Package: castarter (via r-universe)

September 16, 2024

Title Content Analysis Starter Toolkit

Version 0.2.0.9010

Description Consistent approaches for basic web scraping, text mining and word frequency analysis of textual datasets

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Imports magrittr, dplyr, tidyr, purrr, stringr, slider, tidytext, rlang (>= 0.1.2), shiny, processx, attempt, DT, glue, golem, htmltools, bslib, dygraphs, waiter, ggiraph, lubridate, tbl2xts, cicerone, usethis, tibble, DBI, RSQLite, ellipsis, rvest, xml2, cli, dbplyr, PrettyCols, reactable, progress, shinymeta

Suggests testthat (>= 3.0.0), roxygen2, spelling, knitr, rmarkdown, arrow

Config/testthat/edition 3

Depends R (>= 2.10)

URL https://castarter.tadadit.xyz, https://github.com/giocomai/castarter

BugReports https://github.com/giocomai/castarter/issues

Language en-US

VignetteBuilder knitr

Repository https://giocomai.r-universe.dev

RemoteUrl https://github.com/giocomai/castarter

RemoteRef HEAD

RemoteSha c13159106e306e776a65de5d25cda96a8330883d

Contents

casdb_empty_index_id *Empty data frame with the same format as data stored in the* index_id *table*

Description

Empty data frame with the same format as data stored in the index_id table

Usage

```
casdb_empty_index_id
```

Format

A data frame with 0 rows and 3 columns:

id Numeric. Column meant for unique integer identifier corresponding to a unique url

url Character. A url.

type Character. A textual string, by default index.

cass_build_urls *Helps you define the parameters you need for building index urls*

Description

Helps you define the parameters you need for building index urls

Helps you define the parameters you need for building index urls

Usage

```
cass_build_urls()
```

```
cass_build_urls()
```

Value

Nothing, but prints to the console the function call as created in the Shiny app.

Nothing, called for interactive use.

Examples

```
## Not run:
if (interactive) {
  cass_build_urls()
}

## End(Not run)
## Not run:
if (interactive) {
  cass_build_urls()
}

## End(Not run)
```

`cass_combine_into_pattern` *Combines a vector of words into a string to be used for regex matching.*

Description

Combines a vector of words into a string to be used for regex matching.

Usage

```
cass_combine_into_pattern(words, full_words_only = TRUE)
```

Arguments

`words`	A character vector of words to be combined for string matching.
`full_words_only`	Logical, defaults to TRUE. If TRUE, the correspondent words are matched only when they are a separate word.

Value

A character vector of length one, ready to be used for regex matching.

Examples

```
words <- c("dogs", "cats", "horses")

cass_combine_into_pattern(words)
```

cass_download_csv_app *A minimal shiny app that demonstrates the functioning of related modules*

Description

A minimal shiny app that demonstrates the functioning of related modules

Usage

```
cass_download_csv_app(df, type)
```

Arguments

df	A data frame to be exported as csv.

Value

A shiny app

Examples

```
count_df <- castarter::cas_count(
  corpus = castarter::cas_demo_corpus,
  string = c("russia", "moscow")
) %>%
  cas_summarise(before = 15, after = 15)

# cass_cass_download_csv_app(count_df)
```

cass_highlight *Takes a character vector and returns it with matches of pattern wrapped in html tags used for highlighting*

Description

Takes a character vector and returns it with matches of pattern wrapped in html tags used for highlighting

Usage

```
cass_highlight(string, pattern, ignore_case = TRUE)
```

Arguments

string	A character vector.
ignore_case	Defaults to TRUE.
param	Pattern to match.

Examples

```
cass_highlight(
  string = c(
    "The R Foundation for Statistical Computing",
    "R is free software and comes with ABSOLUTELY NO WARRANTY"
  ),
  pattern = "foundation|software|warranty"
)
```

`cass_show_ts_dygraph_app` *A minimal shiny app that demonstrates the functioning of related modules*

Description

A minimal shiny app that demonstrates the functioning of related modules

Usage

```
cass_show_ts_dygraph_app(count_df)
```

Arguments

`count_df`	A dataframe with three columns (`date`, `word`, and `n`), typically created with `cas_count()` and possibly processed with `cas_summarise()`.

Value

A shiny app

Examples

```
count_df <- castarter::cas_count(
  corpus = castarter::cas_demo_corpus,
  string = c("russia", "moscow")
) %>%
  cas_summarise(before = 15, after = 15)

# cass_show_ts_dygraph_app(count_df)
```

cass_split_string *Split string into multiple inputs*

Description

Split string into multiple inputs

Usage

```
cass_split_string(string, squish = TRUE, to_lower = TRUE, to_regex = FALSE)
```

Arguments

string	A text string, typically a user input in a shiny app.
to_regex	Defaults to FALSE. If TRUE collapses the split string, separating each element with |.

Value

A character vector

Examples

```
cass_split("dogs, cats, horses")
cass_split(string = "dogs, cats, horses", to_regex = TRUE)
```

cas_archive *Archive originals of downloaded files in compressed folders*

Description

Archive originals of downloaded files in compressed folders

Usage

```
cas_archive(
  path = NULL,
  file_format = "tar.gz",
  index = TRUE,
  contents = TRUE,
  remove_original = TRUE,
  db_connection = NULL,
  db_folder = NULL,
  ...
)
```

Arguments

path	Path to archive directory, defaults to NULL. If NULL, path is set to the project/website/archive folder.
file_format	Defaults to "tar.gz", to ensure cross-platform compatibility. No other formats are supported at this stage.
remove_original	Defaults to TRUE. If TRUE, after local files have been confirmed to be stored in the relevant compressed file, they are removed from their original folders, and the empty folders deleted.
db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
...	Passed to `cas_get_db_file()`.

---

cas_backup_gd *Backup files to Google Drive*

Description

Backup files to Google Drive

Usage

```
cas_backup_gd(
  glob = c("*.tar.gz", "*.sqlite"),
  email = gargle::gargle_oauth_email(),
  scopes = "https://www.googleapis.com/auth/drive.file",
  client = cas_google_client,
  ...
)
```

Arguments

glob	A character vector with all glob selectors for the type of files to be stored. Defaults to `c("*.tar.gz", "*.sqlite")`.
email	If given, email of the Google account to use for storing files.
scopes	Defaults to `drive.file`, i.e., only give access to files created with this client. This means that no access to other files and folders on your Google Drive is ever given to this session.
client	Google app client, defaults to `castarter`'s own. Passed to `googledrive::drive_auth_configure`. Set to NULL to use `googledrive`'s defaults.
...	

cas_browse *Open in a browser a URL stored in the local database*

Description

This function is typically used to check a web page when extracting links from index, or contents from contents pages.

Usage

```
cas_browse(
  index = FALSE,
  remote = TRUE,
  id = NULL,
  batch = NULL,
  index_group = NULL,
  file_format = "html",
  sample = 1,
  disconnect_db = TRUE,
  ...
)
```

Arguments

index	Logical, defaults to FALSE. If TRUE, downloaded files will be considered `index` files. If not, they will be considered `contents` files. See Readme for a more extensive explanation.
remote	Defaults to TRUE. If TRUE, opens relevant url online. If FALSE, it opens the locally stored file.
sample	Defaults to 1. By default, it opens one random url.
...	Passed to `cas_get_db_file()`.

cas_build_urls *URL builder*

Description

Convenience function typically used to generate urls to index pages listing articles.

Usage

```
cas_build_urls(
  url,
  url_ending = "",
  glue = FALSE,
  start_page = NULL,
  end_page = NULL,
  increase_by = 1,
  date_format = "Ymd",
  start_date = NULL,
  end_date = Sys.Date() - 1,
  date_separator = NULL,
  increase_date_by = "day",
  reversed_order = FALSE,
  index_group = "index",
  index = TRUE,
  write_to_db = FALSE,
  ...
)
```

Arguments

`url`	First part of index link that does not change in other index pages.
`url_ending`	Part of index link appneded after the part of the link that varies. If not relevant, may be left empty.
`glue`	Logical, defaults to FALSE. If TRUE, the url is parsed with `glue`, enabling custom or repeated location for the variable part of the url. If `glue` is set to TRUE, it is expected that the url will include the string `{here}` within curly brackets, e.g. `https://example.com/archive/?from_date={here}&to_date={here}`.
`start_page`	If the urls include a numerical component, define first number of the sequence. Defaults to NULL. If given, coerced to numeric, expected to be an integer.
`end_page`	If the urls include a numerical component, define first number of the sequence. Defaults to NULL. If given, coerced to numeric, expected to be an integer.
`increase_by`	Defines by how much the number in the link should be increased in the numerical sequence. Defaults to 1.
`date_format`	A character string, defaults to "YMD". Check `strptime` for valid values used to define the format of the date that is part of the URL. Simplified formats such as the following are also accepted: "Y" (e.g. 2022), "Ym" (2022-10), "Ymd" (e.g. 2022-10-24). See details.
`start_date`	Defaults to NULL. If given, a date, or a character vector of length one coercible to date with `as.Date`. When given, urls are built based on dates, and parameters `start_page`, `end_page`, and `increase_by`, are ignored.
`end_date`	Defaults to `Sys.Date()`. If given, a date, or a character vector of length one coercible to date with `as.Date`.
`increase_date_by`	Defaults to "day". See `seq.Date` for valid values.

reversed_order	Logical, defaults to FALSE. If TRUE, the order of urls in the output.
index_group	A character vector, defaults to "index". Used for differentiating among different types of index or links in local databases.
index	Defaults to TRUE. Relevant only if write_to_db is also set to TRUE. If TRUE, urls are stored in the local database in the index table, otherwise they are stored in the contents table.
write_to_db	Defaults to FALSE. If set to TRUE, stores the newly created URLs to the local database.

Value

A data frame with three columns, id, url, and index_group. Typically, url corresponds to a vector of unique urls.

Date formats

It is not uncommon in particular for index pages to include dates in the URL, along the lines of example.com/archive/2022-01-01, example.com/archive/2022-01-02, etc. To build such urls, cas_build_urls needs a start_date and end_date. The formatting of the date can be defined either by providing to the parameter date_format a string that strptime is able to interpret directly, or a simplified string (such as "Ymd", without the "%"),adding a date_separator such as "-" as needed.

Examples

```
cas_build_urls(
  url = "https://www.example.com/news/",
  start_page = 1,
  end_page = 10
)

cas_build_urls(
  url = "https://example.com/news/?skip=",
  start_page = 0,
  end_page = 100,
  increase_by = 10
)


cas_build_urls(
  url = "https://example.com/archive/",
  start_date = "2022-01-01",
  end_date = "2022-12-31",
  date_separator = "-"
) %>%
  head()

cas_build_urls(
  url = "https://example.com/archive/?from={here}&to={here}",
  glue = TRUE,
```

```
  start_date = "2011-01-01",
  end_date = "2022-12-31",
  date_separator = ".",
  date_format = "dmY",
  index_group = "news"
)
```

---

cas_check_corpus *Checks if given corpus exists, and, optionally updates it*

Description

Checks if given corpus exists, and, optionally updates it

Usage

```
cas_check_corpus(
  ...,
  update = FALSE,
  keep_only_latest = FALSE,
  path = NULL,
  file_format = "parquet",
  partition = NULL,
  token = "full_text",
  corpus_folder = "corpus"
)
```

Arguments

`...`	Passed to `cas_get_db_file()`.
`update`	Logical, defaults to FALSE. If set to TRUE, it checks if the local database has contents with a higher content id than is currently available in previously exported corpus, if any. If so, it writes a new, updated corpus.
`keep_only_latest`	Logical, defaults to FALSE. If set to TRUE, it deletes previous, older, corpora of the same type.
`path`	Defaults to NULL. If NULL, path is set to the project/website/export/dataset/file_format folder.
`file_format`	Defaults to "parquet". Currently, other options are not implemented.
`partition`	Defaults to NULL. If NULL, the parquet file is not partitioned. "year" is a common alternative: if set to "year", the parquet file is partitioned by year. If a `year` column does not exist, it is created based on the assumption that a `date` column exists and it is (or can be coerced to) a vector of class `Date`.
`token`	Defaults to "full_text", which does not tokenise the text column. If different from `full_text`, it is passed to `tidytext::unnest_tokens` (see its help for details). Accepted values include "words", "sentences", and "paragraphs". See `?tidytext::unnest_tokens()` for details.

Value

Path to corpus. NULL, if no corpus is found and update is set to FALSE.

`cas_check_db_folder` *Checks if database folder exists, if not returns an informative message*

Description

Checks if database folder exists, if not returns an informative message

Usage

```
cas_check_db_folder()
```

Value

If the database folder exists, returns TRUE. Otherwise throws an error.

See Also

Other database functions: `cas_check_use_db()`, `cas_connect_to_db()`, `cas_create_db_folder()`, `cas_disable_db()`, `cas_disconnect_from_db()`, `cas_enable_db()`, `cas_get_db_settings()`, `cas_read_from_db()`, `cas_set_db()`, `cas_set_db_folder()`, `cas_write_to_db()`

Examples

```
# If database folder does not exist, it throws an error
tryCatch(cas_check_db_folder(),
  error = function(e) {
    return(e)
  }
)

# Create database folder
cas_set_db_folder(path = fs::path(
  tempdir(),
  "cas_db_folder"
))
cas_create_db_folder(ask = FALSE)

cas_check_db_folder()
```

cas_check_read_db_contents_data

Returns a corpus from the `contents_data` *table in the database; if corpus is give, it just returns that instead.*

Description

Mostly used internally

Usage

```
cas_check_read_db_contents_data(
  corpus = NULL,
  collect = FALSE,
  db_connection = NULL,
  db_folder = NULL,
  ...
)
```

Arguments

`collect`	Logical, defaults to FALSE. If TRUE, it always returns a data frame and not a database connection, no matter the input.
`...`	Passed to `cas_get_db_file()`.

cas_check_use_db

Check caching status in the current session, and override it upon request

Description

Mostly used internally in functions, exported for reference.

Usage

```
cas_check_use_db(use_db = NULL, ...)
```

Arguments

`use_db`	Defaults to NULL. If NULL, checks current use_db settings. If given, returns given value, ignoring use_db.

Value

Either TRUE or FALSE, depending on current use_db settings.

See Also

Other database functions: cas_check_db_folder(), cas_connect_to_db(), cas_create_db_folder(), cas_disable_db(), cas_disconnect_from_db(), cas_enable_db(), cas_get_db_settings(), cas_read_from_db(), cas_set_db(), cas_set_db_folder(), cas_write_to_db()

Examples

```
cas_check_use_db()
```

---

cas_check_website_folder

Checks if current website folder exists

Description

Parameters can be left to NULL; it will then rely on parameters set with cas_set_options()

Usage

```
cas_check_website_folder(base_folder = NULL, project = NULL, website = NULL)
```

Arguments

base_folder	Defaults to NULL, can be set once per session with cas_set_options(). A path to a location used for storing html and other project files. If the folder does not exist, it will be created. If not given, and not previously set as environment variable, defaults to castarter_data.
project	Defaults to NULL. Project name, can be set once per session with cas_set_options(). This will be used as first level folder and may be used elsewhere to describe the dataset.
website	Defaults to NULL. Website name, can be set once per session with cas_set_options(). This will be used as a second level folder and may be used elsewhere to describe the dataset.

Value

Logical, TRUE if website folder exists, FALSE if it does not.

cas_connect_to_db *Return a connection to be used for caching*

Description

Return a connection to be used for caching

Usage

```
cas_connect_to_db(
  db_connection = NULL,
  use_db = NULL,
  db_type = NULL,
  db_folder = NULL,
  read_only = FALSE,
  ...
)
```

Arguments

db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
use_db	Defaults to NULL. If given, it should be given either TRUE or FALSE. Typically set with cas_enable_db() or cas_disable_db().
read_only	Defaults to FALSE. Passed to DBI::dbConnect.
...	Passed to cas_get_db_file().

Value

A connection object.

See Also

Other database functions: cas_check_db_folder(), cas_check_use_db(), cas_create_db_folder(), cas_disable_db(), cas_disconnect_from_db(), cas_enable_db(), cas_get_db_settings(), cas_read_from_db(), cas_set_db(), cas_set_db_folder(), cas_write_to_db()

Examples

```
if (interactive()) {
  db_connection <- DBI::dbConnect(
    RSQLite::SQLite(), # or e.g. odbc::odbc(),
    Driver = ":memory:", # or e.g. "MariaDB",
    Host = "localhost",
    database = "example_db",
    UID = "example_user",
    PWD = "example_pwd"
```

```
  )
  cas_connect_to_db(db_connection)


  db_settings <- list(
    driver = "MySQL",
    host = "localhost",
    port = 3306,
    database = "castarter",
    user = "secret_username",
    pwd = "secret_password"
  )

  cas_connect_to_db(db_settings)
}
```

`cas_convert_db_type` *Convert database type, e.g. from DuckDB to SQLite*

Description

Convert database type, e.g. from DuckDB to SQLite

Usage

```
cas_convert_db_type(
  source_db_type,
  destination_db_type,
  disconnect_db = FALSE,
  ...
)
```

Arguments

`source_db_type`	A database type, such as "DuckDB" or "SQLite". Must be declared explicitly.
`destination_db_type`	A database type, such as "DuckDB" or "SQLite". Must be declared explicitly.

cas_count *Count strings in a corpus*

Description

Count strings in a corpus

Usage

```
cas_count(
  corpus,
  pattern,
  text = text,
  group_by = date,
  ignore_case = TRUE,
  drop_na = TRUE,
  fixed = FALSE,
  full_words_only = FALSE,
  pattern_column_name = pattern,
  n_column_name = n,
  locale = "en"
)
```

Arguments

corpus	A textual corpus as a data frame.
pattern	A character vector of one or more words or strings to be counted.
text	Defaults to text. The unquoted name of the column of the corpus data frame to be used for matching.
group_by	Defaults to NULL. If given, the unquoted name of the column to be used for grouping (e.g. date, or doc_id, or source, etc.)
ignore_case	Defaults to TRUE.
drop_na	Defaults to TRUE. If TRUE, all rows where either text or group_by column is NA are removed before further processing.
full_words_only	Defaults to FALSE. If FALSE, string is counted even when the it is found in the middle of a word (e.g. if FALSE, "ratio" would be counted as match in the word "irrational").
pattern_column_name	Defaults to word. The unquoted name of the column to be used for the word in the output.
n_column_name	Defaults to n. The unquoted name of the column to be used for the count in the output.
locale	Locale to be used when ignore_case is set to TRUE. Passed to stringr::str_to_lower, defaults to "en".

Value

A data frame

Examples

```
## Not run:
cas_count(
  corpus = corpus,
  pattern = c("dogs", "cats", "horses"),
  text = text,
  group_by = date,
  n_column_name = n
)

## End(Not run)
```

`cas_count_relative` *Count strings in a corpus relative to the number of words*

Description

Count strings in a corpus relative to the number of words

Usage

```
cas_count_relative(
  corpus,
  pattern,
  text = text,
  group_by = date,
  ignore_case = TRUE,
  fixed = FALSE,
  full_words_only = FALSE,
  pattern_column_name = pattern,
  n_column_name = n,
  locale = "en"
)
```

Arguments

`corpus`	A textual corpus as a data frame.
`pattern`	A character vector of one or more words or strings to be counted.
`text`	Defaults to text. The unquoted name of the column of the corpus data frame to be used for matching.
`group_by`	Defaults to NULL. If given, the unquoted name of the column to be used for grouping (e.g. date, or doc_id, or source, etc.)

ignore_case	Defaults to TRUE.
full_words_only	Defaults to FALSE. If FALSE, string is counted even when the it is found in the middle of a word (e.g. if FALSE, "ratio" would be counted as match in the word "irrational").
pattern_column_name	Defaults to 'word'. The unquoted name of the column to be used for the word in the output (if `include_string` is set to TRUE, as per default).
n_column_name	Defaults to 'n'. The unquoted name of the column to be used for the count in the output.
locale	Locale to be used when ignore_case is set to TRUE. Passed to `stringr::str_to_lower`, defaults to "en".

Value

A data frame

Examples

```
## Not run:
cas_count_relative(
  corpus = corpus,
  pattern = c("dogs", "cats", "horses"),
  text = text,
  group_by = date,
  n_column_name = n
)

## End(Not run)
```

---

cas_count_total_words *Count total words in a dataset*

Description

Count total words in a dataset

Usage

```
cas_count_total_words(
  corpus,
  pattern = "\\w+",
  text = text,
  group_by = date,
  ignore_case = TRUE,
  n_column_name = n,
  locale = "en"
)
```

Arguments

corpus	A textual corpus as a data frame.
pattern	Defaults to pattern commonly used to count words.
text	Defaults to text. The unquoted name of the column of the corpus data frame to be used for matching.
group_by	Defaults to NULL. If given, the unquoted name of the column to be used for grouping (e.g. date, or doc_id, or source, etc.)
ignore_case	Defaults to TRUE.
n_column_name	Defaults to n. The unquoted name of the column to be used for the count in the output.
locale	Locale to be used when ignore_case is set to TRUE. Passed to stringr::str_to_lower, defaults to "en".

---

cas_create_db_folder *Creates the base folder where* castarter *stores the project database.*

Description

Creates the base folder where castarter stores the project database.

Usage

```
cas_create_db_folder(path = NULL, ask = TRUE, ...)
```

Arguments

ask	Logical, defaults to TRUE. If FALSE, and database folder does not exist, it just creates it without asking (useful for non-interactive sessions).

Value

Nothing, used for its side effects.

See Also

Other database functions: cas_check_db_folder(), cas_check_use_db(), cas_connect_to_db(), cas_disable_db(), cas_disconnect_from_db(), cas_enable_db(), cas_get_db_settings(), cas_read_from_db(), cas_set_db(), cas_set_db_folder(), cas_write_to_db()

Examples

```
cas_create_db_folder(path = fs::path(fs::path_temp(), "cas_data"))
```

cas_delete_corpus *Delete previously stored corpora written with* cas_write_corpus().

Description

Typically used for file maintainance, especially when datasets are routinely updated.

Usage

```
cas_delete_corpus(
  keep = 1,
  ask = TRUE,
  file_format = "parquet",
  partition = "year",
  token = "full_text",
  corpus_folder = "corpus",
  path = NULL,
  ...
)
```

Arguments

keep	Numeric, defaults to 1. Number of corpus files to keep. Only the most recent files are kept.
file_format	Defaults to "parquet". Currently, other options are not implemented.
partition	Defaults to NULL. If NULL, the parquet file is not partitioned. "year" is a common alternative: if set to "year", the parquet file is partitioned by year. If a year column does not exist, it is created based on the assumption that a date column exists and it is (or can be coerced to) a vector of class Date.
token	Defaults to "full_text", which does not tokenise the text column. If different from full_text, it is passed to tidytext::unnest_tokens (see its help for details). Accepted values include "words", "sentences", and "paragraphs". See ?tidytext::unnest_tokens() for details.
path	Defaults to NULL. If NULL, path is set to the project/website/export/dataset/file_format folder.
...	Passed to cas_get_db_file().

`cas_delete_from_db` *Delete rows from selected database table*

Description

Delete rows from selected database table

Usage

```
cas_delete_from_db(
  table,
  id = NULL,
  batch = NULL,
  index_group = NULL,
  ask = TRUE,
  db_folder = NULL,
  db_connection = NULL,
  disconnect_db = FALSE,
  ...
)
```

Arguments

`table`	Name of the table from where rows should be deleted.
`id`	Defaults to NULL. A vector of id. Rows with the given id will be removed from the database.
`batch`	Defaults to NULL. A vector of batch identigiers. Rows with the given batch id will be removed from the database.
`index_group`	Defaults to NULL. A vector of "index_group" names. Rows with the given "index_group" will be removed from the database.
`ask`	Defaults to TRUE. If TRUE, it runs a query checking how many rows would be deleted, and actually deletes them only after confirming.
`db_connection`	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
`disconnect_db`	Defaults to TRUE. If FALSE, leaves the connection to database open.
`...`	Passed to `cas_get_db_file()`.

Value

Nothing, used for its side effects.

Examples

```
## Not run:
if (interactive) {
  cas_delete_from_db(table = "contents_data", id = id_to_delete)
}

## End(Not run)
```

`cas_disable_db` *Disable caching for the current session*

Description

Disable caching for the current session

Usage

```
cas_disable_db()
```

Value

Nothing, used for its side effects.

See Also

Other database functions: `cas_check_db_folder()`, `cas_check_use_db()`, `cas_connect_to_db()`, `cas_create_db_folder()`, `cas_disconnect_from_db()`, `cas_enable_db()`, `cas_get_db_settings()`, `cas_read_from_db()`, `cas_set_db()`, `cas_set_db_folder()`, `cas_write_to_db()`

Examples

```
if (interactive()) {
  cas_disable_db()
}
```

`cas_disconnect_from_db`
Ensure that connection to database is disconnected consistently

---

Description

Ensure that connection to database is disconnected consistently

Usage

```
cas_disconnect_from_db(
  db_connection = NULL,
  db_type = NULL,
  use_db = NULL,
  disconnect_db = FALSE
)
```

Arguments

db_connection	Defaults to NULL. If NULL, and database is enabled, castarter will use a local sqlite database. A custom connection to other databases can be given (see vignette castarter_db_management for details).
use_db	Defaults to NULL. If given, it should be given either TRUE or FALSE. Typically set with cas_enable_db() or cas_disable_db().
disconnect_db	Defaults to TRUE. If FALSE, leaves the connection to database open.

Value

Nothing, used for its side effects.

See Also

Other database functions: cas_check_db_folder(), cas_check_use_db(), cas_connect_to_db(), cas_create_db_folder(), cas_disable_db(), cas_enable_db(), cas_get_db_settings(), cas_read_from_db(), cas_set_db(), cas_set_db_folder(), cas_write_to_db()

Examples

```
cas_disconnect_from_db()
```

---

cas_download *Downloads files systematically, and stores details about the download in a local database*

Description

Downloads files systematically, and stores details about the download in a local database

Usage

```
cas_download(
  download_df = NULL,
  index = FALSE,
  index_group = NULL,
  file_format = "html",
  overwrite_file = FALSE,
```

```
  create_folder_if_missing = NULL,
  ignore_id = TRUE,
  wait = 1,
  pause_base = 2,
  pause_cap = 256,
  pause_min = 4,
  sample = FALSE,
  retry_times = 3,
  terminate_on = NULL,
  user_agent = NULL,
  download_again_if_status_is_not = NULL,
  ...
)
```

Arguments

index	Logical, defaults to FALSE. If TRUE, downloaded files will be considered index files. If not, they will be considered contents files. See Readme for a more extensive explanation.
overwrite_file	Logical, defaults to FALSE. If TRUE, files are downloaded again even if already present, overwriting previously downloaded items.
wait	Defaults to 1. Number of seconds to wait between downloading one page and the next. Can be increased to reduce server load, or can be set to 0 when this is not an issue.
sample	Defaults to FALSE. If TRUE, the download order is randomised. If a numeric is given, the download order is randomised and at most the given number of items is downloaded.
retry_times	Defaults to 3. Number of times to retry download in case of errors.
user_agent	Defaults to NULL. If given, passed to download method.
...	Passed to cas_get_db_file().
urls_df	A data frame with at least two columns named id and url. Typically generated with cas_build_urls() for index files. If a character vector is given instead, identifiers will be given automatically.

---

cas_download_chromote *Downloads one file at a time with chromote*

Description

Downloads one file at a time with chromote

Usage

```
cas_download_chromote(
  download_df = NULL,
  index = FALSE,
  index_group = NULL,
  overwrite_file = FALSE,
  ignore_id = TRUE,
  wait = 1,
  db_connection = NULL,
  sample = FALSE,
  file_format = "html",
  disconnect_db = FALSE,
  ...
)
```

Arguments

download_df	A data frame with four columns: id, url, path, type.
index	Logical, defaults to FALSE. If TRUE, downloaded files will be considered index files. If not, they will be considered contents files. See Readme for a more extensive explanation.
overwrite_file	Logical, defaults to FALSE.
wait	Defaults to 1. Number of seconds to wait between downloading one page and the next. Can be increased to reduce server load, or can be set to 0 when this is not an issue.
db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
sample	Defaults to FALSE. If TRUE, the download order is randomised. If a numeric is given, the download order is randomised and at most the given number of items is downloaded.
disconnect_db	Defaults to TRUE. If FALSE, leaves the connection to database open.
...	Passed to cas_get_db_file().

---

cas_download_httr *Downloads one file at a time with httr*

Description

Mostly used internally by cas_download.

Usage

```
cas_download_httr(
  download_df = NULL,
  index = FALSE,
  index_group = NULL,
  overwrite_file = FALSE,
  ignore_id = TRUE,
  wait = 1,
  create_folder_if_missing = NULL,
  pause_base = 2,
  pause_cap = 256,
  pause_min = 4,
  terminate_on = NULL,
  retry_times = 3,
  db_connection = NULL,
  disconnect_db = FALSE,
  sample = FALSE,
  file_format = "html",
  user_agent = NULL,
  download_again_if_status_is_not = NULL,
  ...
)
```

Arguments

download_df	A data frame with four columns: id, url, path, type.
index	Logical, defaults to FALSE. If TRUE, downloaded files will be considered index files. If not, they will be considered contents files. See Readme for a more extensive explanation.
overwrite_file	Logical, defaults to FALSE.
wait	Defaults to 1. Number of seconds to wait between downloading one page and the next. Can be increased to reduce server load, or can be set to 0 when this is not an issue.
retry_times	Defaults to 3. Number of times to retry download in case of errors.
db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
disconnect_db	Defaults to TRUE. If FALSE, leaves the connection to database open.
sample	Defaults to FALSE. If TRUE, the download order is randomised. If a numeric is given, the download order is randomised and at most the given number of items is downloaded.
user_agent	Defaults to NULL. If given, passed to download method.
...	Passed to cas_get_db_file().

Value

Invisibly returns the full httr response.

cas_download_index *Downloads index files systematically, and stores details about the download in a local database*

Description

Downloads index files systematically, and stores details about the download in a local database

Usage

```
cas_download_index(
  download_df = NULL,
  index_group = NULL,
  file_format = "html",
  overwrite_file = FALSE,
  create_folder_if_missing = NULL,
  wait = 1,
  pause_base = 2,
  pause_cap = 256,
  pause_min = 4,
  sample = FALSE,
  retry_times = 8,
  terminate_on = 404,
  user_agent = NULL,
  download_again_if_status_is_not = NULL,
  ...
)
```

Arguments

index

cas_download_internal *Downloads one file at a time with readLines*

Description

Mostly used internally by cas_download.

Usage

```
cas_download_internal(
  download_df = NULL,
  index = FALSE,
  index_group = NULL,
```

```
  overwrite_file = FALSE,
  ignore_id = TRUE,
  wait = 1,
  create_folder_if_missing = NULL,
  db_connection = NULL,
  disconnect_db = FALSE,
  sample = FALSE,
  file_format = "html",
  ...
)
```

Arguments

download_df	A data frame with four columns: id, url, path, type.
index	Logical, defaults to FALSE. If TRUE, downloaded files will be considered index files. If not, they will be considered contents files. See Readme for a more extensive explanation.
overwrite_file	Logical, defaults to FALSE.
wait	Defaults to 1. Number of seconds to wait between downloading one page and the next. Can be increased to reduce server load, or can be set to 0 when this is not an issue.
db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
disconnect_db	Defaults to TRUE. If FALSE, leaves the connection to database open.
sample	Defaults to FALSE. If TRUE, the download order is randomised. If a numeric is given, the download order is randomised and at most the given number of items is downloaded.
...	Passed to cas_get_db_file().

Value

Invisibly returns the full httr response.

---

cas_download_legacy *Downloads html pages based on a vector of links*

Description

Downloads html pages based on a vector of links.

Usage

```
cas_download_legacy(
  url,
  type = "contents",
  custom_folder = NULL,
  custom_path = NULL,
  file_format = "html",
  url_to_download = NULL,
  size = 500,
  wget_system = FALSE,
  method = "auto",
  missing_pages = TRUE,
  start = 1,
  wait = 1,
  ignore_ssl_certificates = FALSE,
  use_headless_chromium = FALSE,
  headless_chromium_wait = 1,
  use_phantomjs = FALSE,
  create_script = FALSE,
  project = NULL,
  website = NULL,
  base_folder = NULL
)
```

Arguments

`url`	A character vector of urls, or a data frame with at least two columns named `id` and `url`.
`type`	Accepted values are either "contents" (default), "index".
`custom_folder`	Defaults to NULL. If given, overrides the "type" param and stores files in given path as a subfolder of project/website. Folder must already exist, and should be empty.
`url_to_download`	Defaults to NULL. If given, expected to be a logical vector to be applied to the given urls. If given, it takes precedence over `missing_pages` and `size`.
`size`	Defaults to 500. It represents the minimum size in bytes that downloaded html files should have: files that are smaller will be downloaded again. Used only when missing_pages == FALSE.
`wget_system`	Logical, defaults to FALSE. Calls wget as a system command through the system() function. Wget must be previously installed on the system.
`method`	Defaults to "auto". Method is passed to the function utils::download.file(); available options are "internal", "wininet" (Windows only) "libcurl", "wget" and "curl". For more information see ?utils::download.file()
`missing_pages`	Logical, defaults to TRUE. If TRUE, verifies if a downloaded html file exists for each element in articlesLinks; when there is no such file, it downloads it.

start Integer. Only url with position higher than start in the url vector will be downloaded: `url[start:length(url)]`

ignore_ssl_certificates
Logical, defaults to FALSE. If TRUE it uses wget to download the page, and does not check if the SSL certificate is valid. Useful, for example, for https pages with expired or mis-configured SSL certificate.

use_headless_chromium
Logical, defaults to FALSE. If TRUE uses the `crrri` package to download pages. Useful in particular when web pages are generated via javascript. See in particular: https://github.com/RLesur/crrri#system-requirements

headless_chromium_wait
Numeric, in seconds. How long should headless chrome wait after loading page?

create_script Logical, defaults to FALSE. Tested on Linux only. If TRUE, creates a downloadPages.sh executable file that can be used to download all relevant pages from a terminal.

project Name of 'castarter2' project. Must correspond to the name of a folder in the current working directory.

website Name of a website included in a 'castarter2' project. Must correspond to the name of a sub-folder of the project folder.

path Defaults to NULL. If given, overrides the "type" and "custom_folder" param and stores files in given path.

Value

By default, returns nothing, used for its side effects (downloads html files in relevant folder). Download files can then be imported in a vector with the function ImportHtml.

Examples

```
## Not run:
if (interactive()) {
  cas_download(url)
}

## End(Not run)
```

---

cas_enable_db *Enable caching for the current session*

Description

Enable caching for the current session

Usage

```
cas_enable_db(db_type = "SQLite")
```

Value

Nothing, used for its side effects.

See Also

Other database functions: cas_check_db_folder(), cas_check_use_db(), cas_connect_to_db(), cas_create_db_folder(), cas_disable_db(), cas_disconnect_from_db(), cas_get_db_settings(), cas_read_from_db(), cas_set_db(), cas_set_db_folder(), cas_write_to_db()

Examples

```
if (interactive()) {
  cas_enable_db()
}
```

---

cas_explorer *Run the Shiny Application*

Description

Run the Shiny Application

Usage

```
cas_explorer(
  corpus = castarter::cas_demo_corpus,
  default_pattern = NULL,
  title = "castarter",
  collect = FALSE,
  advanced = FALSE,
  custom_head_html = "<meta name=\"referrer\" content=\"no-referrer\" />",
  footer_html = shiny::tagList(),
  onStart = NULL,
  options = list(),
  enableBookmarking = NULL,
  uiPattern = "/",
  ...
)
```

Arguments

collect	Defaults to FALSE. If TRUE, retrieves the corpus in memory, even if is originally read from a parquet file or a database. With arrow version before 14 that do not have full support of using stringr in context, setting this to TRUE is probably advisable (currently, depending on the arrow version, there may be issues where upper/lower case is not ignored).

custom_head_html
: Chunk of code to be included in the app's <head>. This can be used, e.g., for custom analytics snippets. The default value, <meta name="referrer" content="no-referrer" /> asks the browser not to include the source website when following links to external websites.

onStart
: A function that will be called before the app is actually run. This is only needed for shinyAppObj, since in the shinyAppDir case, a global.R file can be used for this purpose.

options
: Named options that should be passed to the runApp call (these can be any of the following: "port", "launch.browser", "host", "quiet", "display.mode" and "test.mode"). You can also specify width and height parameters which provide a hint to the embedding environment about the ideal height/width for the app.

enableBookmarking
: Can be one of "url", "server", or "disable". The default value, NULL, will respect the setting from any previous calls to enableBookmarking(). See enableBookmarking() for more information on bookmarking your app.

uiPattern
: A regular expression that will be applied to each GET request to determine whether the ui should be used to handle the request. Note that the entire request path must match the regular expression in order for the match to be considered successful.

...
: arguments to pass to golem_opts. See ?golem::get_golem_options for more details.

---

cas_explorer_legacy *Run the Shiny Application*

Description

Run the Shiny Application

Usage

```
cas_explorer_legacy(
  corpus = castarter::cas_demo_corpus,
  default_string = NULL,
  custom_head_html = "<meta name=\"referrer\" content=\"no-referrer\" />",
  onStart = NULL,
  options = list(),
  enableBookmarking = NULL,
  uiPattern = "/",
  ...
)
```

Arguments

onStart	A function that will be called before the app is actually run. This is only needed for shinyAppObj, since in the shinyAppDir case, a global.R file can be used for this purpose.
options	Named options that should be passed to the runApp call (these can be any of the following: "port", "launch.browser", "host", "quiet", "display.mode" and "test.mode"). You can also specify width and height parameters which provide a hint to the embedding environment about the ideal height/width for the app.
enableBookmarking	Can be one of "url", "server", or "disable". The default value, NULL, will respect the setting from any previous calls to enableBookmarking(). See enableBookmarking() for more information on bookmarking your app.
uiPattern	A regular expression that will be applied to each GET request to determine whether the ui should be used to handle the request. Note that the entire request path must match the regular expression in order for the match to be considered successful.
...	arguments to pass to golem_opts. See ?golem::get_golem_options for more details.

---

cas_export_tables *Export database tables to another format such as csv*

Description

Export database tables to another format such as csv

Usage

```
cas_export_tables(
  path = NULL,
  file_format = "csv.gz",
  tables = NULL,
  db_connection = NULL,
  disconnect_db = FALSE,
  db_folder = NULL,
  ...
)
```

Arguments

path	Defaults to NULL. If NULL, path is set to the project/website/export/file_format folder.
file_format	Defaults to "csv.gz", i.e. compressed csv files. All formats supported by readr::write_csv() are valid.

tables	Defaults to NULL. If NULL, all database tables are exported. If given, names of the database tables to export. See `vignette("castarter-database")` for details.
db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
disconnect_db	Defaults to TRUE. If FALSE, leaves the connection to database open.
...	Passed to `cas_get_db_file()`.

Examples

```
## Not run:
if (interactive) {
  cas_export_tables(file_format = "csv")
}

## End(Not run)
```

---

cas_extract *Extract fields and contents from downloaded files*

Description

Extract fields and contents from downloaded files

Usage

```
cas_extract(
  extractors,
  post_processing = NULL,
  id = NULL,
  ignore_id = TRUE,
  custom_path = NULL,
  index = FALSE,
  store_as_character = TRUE,
  check_previous = TRUE,
  db_connection = NULL,
  file_format = "html",
  sample = FALSE,
  write_to_db = FALSE,
  keep_if_status = 200,
  encoding = "UTF-8",
  readability = FALSE,
  ...
)
```

Arguments

extractors	A named list of functions. See examples for details.
post_processing	Defaults to NULL. If given, it must be a function that takes a data frame as input (logically, a row of the dataset) and returns it with additional or modified columns.
id	Defaults to NULL, identifiers to process when extracting. If given, must be a numeric vector, logically corresponding to the identifiers in the id column, e.g. as returned by cas_read_db_contents_id()
ignore_id	Defaults to TRUE. If TRUE, it checks if identifiers have been added to the local ignore list, typically with cas_ignore_id(), and as retrieved with cas_read_db_ignore_id(). It can also be a numeric vector of identifiers: the given identifiers will not be processed. If FALSE, items will be processed normally.
index	Logical, defaults to FALSE. If TRUE, downloaded files will be considered index files. If not, they will be considered contents files. See Readme for a more extensive explanation.
store_as_character	Logical, defaults to TRUE. If TRUE, it converts to character all extracted contents before writing them to database. This reduces issues of type conversions with the default database backend (for example, SQLite automatically converts dates to numeric) or using different backends. This implies you will need to set data types when you read the database, but it also means that you can consistently expect all columns to be character vectors, which in one form or another are consistently implemented across database backends. Set to FALSE if you want to remain in control of column types.
check_previous	Logical, defaults to TRUE. If FALSE, no check will be conducted to verify if the same content had been previously extracted. If FALSE, write_to_db must be set (or will be set) to FALSE, to prevent duplication of data.
sample	Defaults to FALSE. If TRUE, the download order is randomised. If a numeric is given, the download order is randomised and at most the given number of items is downloaded.
keep_if_status	Defaults to 200. Keep only if recorded download status matches the given status.
...	Passed to cas_get_db_file().

Examples

```
## Not run:
if (interactive) {
  ### Post-processing example ####
  # For example, in order to add a column called `internal_id`
  # that takes the ending digits of the url (assuming the url ends with digits)
  # a function such as the following would be passed to cas_extract
  pp <- function(df) {
    df |>
      dplyr::mutate(internal_id = stringr::str_extract(url, "[[:digit:]]+$"))
  }
}
```

```
cas_extract(
  extractors = extractors_l, # assuming it has already been set
  post_processing = pp
)

## End(Not run)
```

`cas_extract_html` *Facilitates extraction of contents from an html file*

Description

Facilitates extraction of contents from an html file

Usage

```
cas_extract_html(
  html_document,
  container = NULL,
  container_class = NULL,
  container_id = NULL,
  container_name = NULL,
  container_property = NULL,
  container_itemprop = NULL,
  container_instance = NULL,
  attribute = NULL,
  sub_element = NULL,
  no_children = NULL,
  trim = TRUE,
  squish = FALSE,
  no_match = "",
  exclude_css_path = NULL,
  exclude_xpath = NULL,
  custom_xpath = NULL,
  custom_css_path = NULL,
  keep_everything = FALSE,
  extract_text = TRUE,
  as_character = TRUE
)
```

Arguments

`html_document`	An html document parsed with `xml2::read_html()` or `rvest::read_html()`.
`container`	Defaults to NULL. Type of html container from where links are to be extracted, such as "div", "ul", and others. Either `container_class` or `container_id` must also be provided.

container_class
: Defaults to NULL. If provided, also container must be given (and container_id must be NULL). Only text found inside the provided combination of container/class will be extracted.

container_id
: Defaults to NULL. If provided, also container must be given (and container_id must be NULL). Only text found inside the provided combination of container/class will be extracted.

container_itemprop
: Defaults to NULL. If provided, also container must be given (and container_id and container_class must be NULL or will be silently ignored). Only text found inside the provided combination of container/itemprop will be extracted.

container_instance
: Defaults to NULL. If given, it must be an integer. If a given combination is found more than once in the same page, the relevant occurrence is kept. Use with caution, as not all pages always include the same number of elements of the same class/with the same id.

attribute
: Defaults to NULL. If given, type of attribute to extract. Typically used in combination with container, as in cas_extract_html(container = "time", attribute = "datetime").

sub_element
: Defaults to NULL. If provided, also container must be given. Only text within elements of given type under the chosen combination of container/containerClass will be extracted. When given, it will tipically be "p", to extract all p elements inside the selected div.

no_children
: Defaults to FALSE, i.e. by default all subelements of the selected combination (e.g. div with given class) are extracted. If TRUE, only text found under the given combination (but not its subelements) will be extracted. Corresponds to the xpath string /node()[not(self::div)].

trim
: Defaults to TRUE. If TRUE, applies stringr::str_trim() to output, removing whitespace from start and end of string.

squish
: Defaults to FALSE. If TRUE, applies stringr::str_squish() to output, removing whitespace from start and end of string, and replacing any whitespace (including new lines) with a single space.

no_match
: Defaults to "". A common alternative would be NA. Value to return when the given container, selector or element is not found.

exclude_css_path
: Defaults to NULL. To remove script, for example, use script, which is transformed to :not(script). May cause issues, use with caution.

exclude_xpath
: Defaults to NULL. A common pattern when extracting text would be //script|//iframe|//img|//sty as it is assumed that these containers (javascript contents, iframes, css blocks, and images) are most likely undesirable when extracting text. Customise as needed. For example, if besides the above you also want to remove a div of class related-articles, you may use //script|//iframe|//img|//div[@class='related-articles']I careful when using exclude_xpath as the relevant Xpath is removed from the original objext passed to cas_extract_html(). To be clear, the input object is changed, and, for example, if used once in one of the extractors these containers won't be available to other extractors.

custom_xpath Defaults to NULL. If given, all other parameters are ignored and given Xpath used instead.

custom_css_path Defaults to NULL. If given, all other parameters are ignored and given CSSpath used instead.

keep_everything Defaults to FALSE. If TRUE, all text included in the page is returned as a single string.

extract_text Defaults to TRUE. If TRUE, text is extracted.

as_character Defaults to TRUE. If FALSE, and if `extract_text` is set to FALSE, then an `xml_nodeset` object is returned.

Value

A character vector of length one.

Examples

```
## Not run:
if (interactive()) {
  url <- "https://example.com"
  html_document <- rvest::read_html(x = url)

  # example for a tag that looks like:
  # <meta name="twitter:title" content="Example title" />

  cas_extract_html(
    html_document = html_document,
    container = "meta",
    container_name = "twitter:title",
    attribute = "content"
  )


  # example for a tag that looks like:
  # <meta name="keywords" content="various;keywords;">
  cas_extract_html(
    html_document = html_document,
    container = "meta",
    container_name = "keywords",
    attribute = "content"
  )

  # example for a tag that looks like:
  # <meta property="article:published_time" content="2016-10-29T13:09+03:00"/>
  cas_extract_html(
    html_document = html_document,
    container = "meta",
    container_property = "article:published_time",
    attribute = "content"
  )
```

```
}

## End(Not run)
```

`cas_extract_links` *Extract direct links to individual content pages from index pages*

Description

Extract direct links to individual content pages from index pages

Usage

```
cas_extract_links(
  id = NULL,
  batch = "latest",
  domain = NULL,
  index = TRUE,
  index_group = NULL,
  output_index = FALSE,
  output_index_group = NULL,
  include_when = NULL,
  exclude_when = NULL,
  container = NULL,
  container_class = NULL,
  container_id = NULL,
  custom_xpath = NULL,
  custom_css = NULL,
  match = NULL,
  min_length = NULL,
  max_length = NULL,
  attribute_type = "href",
  append_string = NULL,
  remove_string = NULL,
  write_to_db = FALSE,
  file_format = "html",
  keep_only_within_domain = TRUE,
  sample = FALSE,
  check_previous = TRUE,
  check_again = FALSE,
  encoding = "UTF-8",
  reverse_order = FALSE,
  db_connection = NULL,
  disconnect_db = TRUE,
  ...
)
```

Arguments

id	Defaults to NULL. If provided, it should be a vector of integers. Only html files corresponding to given id will be processed.
domain	Defaults to "". Web domain of the website. It is added at the beginning of each link found. If links in the page already include the full web address this should be ignored.
output_index	Defaults to FALSE. If FALSE, new links are added to the contents table. If TRUE, the links extracted will be stored again as index, using output_index_group as index_group.
output_index_group	Defaults to NULL. Relevant only when output_index is set to TRUE. Used to store new index urls in the database with reference to the appropriate group.
include_when	Part of URL found only in links of individual articles to be downloaded. If more than one provided, it includes all links that contains either of the strings provided.
exclude_when	If an URL includes this string, it is excluded from the output. One or more strings may be provided.
container	Defaults to NULL. Type of html container from where links are to be extracted, such as "div", "ul", and others. Either container_class or container_id must also be provided.
container_class	Defaults to NULL. If provided, also container must be given (and container_id must be NULL). Only text found inside the provided combination of container/class will be extracted.
container_id	Defaults to NULL. If provided, also container must be given (and container_id must be NULL). Only text found inside the provided combination of container/class will be extracted.
custom_xpath	Defaults to NULL. If given, all other parameters are ignored and given Xpath used instead.
match	Defaults to NULL. Used when extracting json files. Name of property from where url is to be extracted. N.B. Only partly implemented, please report issues along with specific example where it emerged.
min_length	If a link is shorter than the number of characters given in min_length, it is excluded from the output.
max_length	If a link is longer than the number of characters given in max_length, it is excluded from the output.
attribute_type	Defaults to "href". Type of attribute to extract from links.
append_string	If provided, appends given string to the extracted articles. Typically used to create links for print or mobile versions of the extracted page.
remove_string	If provided, remove given string (or strings) from links.
write_to_db	Logical, defaults to FALSE. If TRUE stored newly extracted links in the database, associates each of them with an id, and records the source for each link.

keep_only_within_domain
: Logical, defaults to TRUE. If TRUE, and domain given, links to external websites are dropped.

check_previous
: Defaults to TRUE. If TRUE, checks if newly found links are previously stored in database, and if they are, it discards them. If FALSE, and write_to_db is also set to FALSE, it does not check for previously stored links.

check_again
: Defaults to FALSE. If FALSE, files from where are at least a link has been extracted are not re-processed. If TRUE, they are processed again. By default, only new links are then actually included in the output or stored in the local database.

reverse_order
: Logical, defaults to FALSE. If TRUE, index files are processed in reverse order of id and batch, which may give more meaningful order to content id. The difference is ultimately cosmetic, and has no substantive impact either way.

db_connection
: Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).

disconnect_db
: Defaults to TRUE. If FALSE, leaves the connection to database open.

...
: Passed to cas_get_db_file().

Value

A data frame.

Examples

```
## Not run:
links <- cas_extract_links(domain = "http://www.example.com/")

## End(Not run)
```

---

cas_extract_script *Extracts scripts from an html page*

Description

Extracts scripts from an html page

Usage

```
cas_extract_script(
  html_document,
  script_type = NULL,
  match = NULL,
  accessors = NULL,
  remove_from_script = NULL
)
```

Arguments

html_document	An html document parsed with xml2::read_html() or rvest::read_html().
script_type	Defaults to NULL. Type of script. Common script types include application/ld+json, text/template, etc.
match	Default to NULL. If given, used to filter extracted scripts. Must be a named vector in the format c(@type = "NewsArticle") for a script of type "NewsArticle".
accessors	Defaults to NULL. If given, a vector of accessors passed to purrr::pluck in order to extract sub-components of the list resulting from reading the with jsonlite the result of the previous steps and filter.
remove_from_script	Defaults to NULL. If given, removed after the script has been extracted but before processing the json.

Value

May return a list or a character vector. If no match is found, returns NA_character_

Examples

```
## Not run:
if (interactive()) {
 url <- "https://www.digi24.ro/stiri/externe/casa-alba-pune-capat-isteriei-globale-nu-exista-indicii-ca-obiecte

  html_document <- rvest::read_html(x = url)

  cas_extract_script(
    html_document = html_document,
    script_type = "application/ld+json"
  )

  # get date published
  cas_extract_script(
    html_document = html_document,
    script_type = "application/ld+json",
    match = c(`@type` = "NewsArticle"),
    accessors = "datePublished"
  )

  # get title
  cas_extract_script(
    html_document = html_document,
    script_type = "application/ld+json",
    match = c(`@type` = "NewsArticle"),
    accessors = "headline"
  )

  # get nested element, e.g. url of the logo of the publisher

  cas_extract_script(
```

```
    html_document = html_document,
    script_type = "application/ld+json",
    match = c(`@type` = "NewsArticle"),
    accessors = c("publisher", "logo", "url")
  )
}

## End(Not run)
```

`cas_find_extractor` *Facilitate finding extractors, typically to be used with* `cas_extract_html()`

Description

This may or may not work, but it may be worth giving this a quick a try before looking for alternatives. The parameters returned first should work best.

Usage

```
cas_find_extractor(
  html_document,
  pattern,
  containers = c("h1", "h2", "h3", "h4", "span", "td", "p", "div"),
  exclude_css_path = NULL
)
```

Arguments

`html_document`	An html document parsed with `xml2::read_html()` or `rvest::read_html()`.
`pattern`	A text string to be matched.
`containers`	Containers to be parsed for best matches. By default: `c("h1", "h2", "h3", "h4", "span", "td", "p", "div")`. The order matters, as results are returned in this order (e.g. if a match of the same length is found both in a "h1" and in a "div", "h1" is returned first).
`exclude_css_path`	Defaults to NULL. To remove script, for example, use `script`, which is transformed to `:not(script)`. May cause issues, use with caution.

Value

A data frame list with container and class or id of values that should work if passed to `cas_extract_html()`.

Examples

```
## Not run:
if (interactive) {
  # not ideal example, but you'll get the gist, see additonal example below
  library("castarter")
 url <- "https://www.nasa.gov/news-release/nasa-sets-coverage-for-roscosmos-spacewalk-outside-space-station/"

  html_page <- rvest::read_html(url)

  cas_find_extractor(
    html_document = html_page,
    pattern = "NASA Sets Coverage for Roscosmos Spacewalk Outside Space Station"
  )

  cas_find_extractor(
    html_document = html_page,
    pattern = "Oct 23, 2023"
  )

  cas_find_extractor(
    html_document = html_page,
    pattern = "Roxana Bardan"
  )

  cas_find_extractor(
    html_document = html_page,
    pattern = "RELEASE"
  )

  ## Use this information to extract contents


  library("castarter")
  url <- "https://www.state.gov/designating-russian-virtual-currency-money-launderer/"

  html_page <- rvest::read_html(url)

  cas_find_extractor(
    html_document = html_page,
    pattern = "Designating Russian Virtual Currency Money Launderer"
  )

  cas_extract_html(
    html_document = html_page,
    container = "span",
    container_class = "bc_current collapse"
  )

  cas_extract_html(
    html_document = html_page,
    container = "h1",
    container_class = "featured-content__headline stars-above"
```

```
  )


  cas_find_extractor(
    html_document = html_page,
    pattern = "Press Statement"
  )

  cas_extract_html(
    html_document = html_page,
    container = "p",
    container_class = "article-meta doctype-meta"
  )


  cas_find_extractor(
    html_document = html_page,
    pattern = "Matthew Miller, Department Spokesperson"
  )

  cas_extract_html(
    html_document = html_page,
    container = "p",
    container_class = "article-meta__author-bureau"
  )

  cas_find_extractor(
    html_document = html_page,
    pattern = "November 3, 2023"
  )

  cas_extract_html(
    html_document = html_page,
    container = "p",
    container_class = "article-meta__publish-date"
  )

  cas_find_extractor(
    html_document = html_page,
    pattern = "The United States is sanctioning Ekaterina Zhdanova",
    exclude_css_path = "script"
  )


  cas_extract_html(
    html_document = html_page,
    container = "div",
    container_class = "entry-content",
    exclude_css_path = "script"
  )
}

## End(Not run)
```

cas_generate_metadata *Generate basic metadata about the corpus, including start and end date and total number of items available.*

Description

Generate basic metadata about the corpus, including start and end date and total number of items available.

Usage

```
cas_generate_metadata(
  corpus = NULL,
  db_connection = NULL,
  db_folder = NULL,
  ...
)
```

Arguments

...	Passed to cas_get_db_file().

Value

A list.

cas_get_base_folder *Get base folder under which files will be stored.*

Description

Get base folder under which files will be stored.

Usage

```
cas_get_base_folder(..., level = "website", custom_path = NULL)
```

Arguments

...	Passed to cas_get_options().
level	Defaults to "website". Valid values are "website", "project", and "base".
custom_path	Defaults to NULL. If given, all other parameters and settings are ignored, and folder is set to this value.

cas_get_base_path *Build full path to base working folder*

Description

Build full path to base working folder

Usage

```
cas_get_base_path(
  create_folder_if_missing = NULL,
  custom_path = NULL,
  custom_folder = NULL,
  index = FALSE,
  file_format = "html",
  ...
)
```

Arguments

create_folder_if_missing	Logical, defaults to NULL. If NULL, it will ask before creating a new folder. If TRUE, it will create it without asking.
custom_path	Defaults to NULL. If given, all other parameters and settings are ignored, and folder is set to this value.
file_format	
...	Passed to cas_get_options().

Value

Path to base folder. A character vector of length one of class fs_path.

cas_get_corpus_path *Get path to folder where the corpus is stored.*

Description

Get path to folder where the corpus is stored.

Usage

```
cas_get_corpus_path(
  ...,
  corpus_folder = "corpus",
  file_format = "parquet",
  partition = NULL,
  token = "full_text"
)
```

Arguments

...	Passed to `cas_get_db_file()`.
file_format	Defaults to "parquet". Currently, other options are not implemented.
partition	Defaults to NULL. If NULL, the parquet file is not partitioned. "year" is a common alternative: if set to "year", the parquet file is partitioned by year. If a `year` column does not exist, it is created based on the assumption that a `date` column exists and it is (or can be coerced to) a vector of class `Date`.
token	Defaults to "full_text", which does not tokenise the text column. If different from `full_text`, it is passed to `tidytext::unnest_tokens` (see its help for details). Accepted values include "words", "sentences", and "paragraphs". See `?tidytext::unnest_tokens()` for details.

Examples

```
## Not run:
cas_get_corpus_path()

## End(Not run)
```

---

cas_get_db *Get connection to database with details about current website*

Description

Get connection to database with details about current website

Usage

```
cas_get_db(
  db_folder = NULL,
  base_folder = NULL,
  project = NULL,
  website = NULL
)
```

Arguments

db_folder	Defaults to NULL. can be set once per session with cas_set_options() or cas_set_db_folder(). A path to a location used for storing the database. If not given, and not previously set as environment variable, defaults to castarter_data.
base_folder	Defaults to NULL, can be set once per session with cas_set_options(). A path to a location used for storing html and other project files. If the folder does not exist, it will be created. If not given, and not previously set as environment variable, defaults to castarter_data.
project	Defaults to NULL. Project name, can be set once per session with cas_set_options(). This will be used as first level folder and may be used elsewhere to describe the dataset.
website	Defaults to NULL. Website name, can be set once per session with cas_set_options(). This will be used as a second level folder and may be used elsewhere to describe the dataset.

Examples

```
cas_get_db(
  base_folder = fs::path_temp(),
  project = "example_project",
  website = "example_website"
)
```

---

cas_get_db_file *Gets location of database file*

Description

Gets location of database file

Usage

```
cas_get_db_file(db_folder = NULL, ...)
```

Value

A character vector of length one with location of the SQLite database file.

Examples

```
cas_set_db_folder(path = tempdir())
db_file_location <- cas_get_db_file(project = "test-project") # outputs location of database file
db_file_location
```

cas_get_db_settings *Get database connection settings from the environment*

Description

Typically set with `cas_set_db()`

Usage

```
cas_get_db_settings()
```

Value

A list with all database parameters as stored in environment variables.

See Also

Other database functions: `cas_check_db_folder()`, `cas_check_use_db()`, `cas_connect_to_db()`, `cas_create_db_folder()`, `cas_disable_db()`, `cas_disconnect_from_db()`, `cas_enable_db()`, `cas_read_from_db()`, `cas_set_db()`, `cas_set_db_folder()`, `cas_write_to_db()`

Examples

```
cas_get_db_settings()
```

cas_get_files_to_download *Create a data frame with not yet downloaded files*

Description

Create a data frame with not yet downloaded files

Usage

```
cas_get_files_to_download(
  urls = NULL,
  index = FALSE,
  index_group = NULL,
  ignore_id = TRUE,
  desc_id = FALSE,
  batch = NULL,
  create_folder_if_missing = NULL,
  custom_folder = NULL,
  custom_path = NULL,
  file_format = "html",
```

```
  db_connection = NULL,
  download_again = FALSE,
  download_again_if_status_is_not = NULL,
  ...
)
```

Arguments

urls	Defaults to NULL. If given, it should correspond with a data frame with at least two columns named id and url. If not given, an attempt will be made to load it from the local database.
index	Logical, defaults to FALSE. If TRUE, downloaded files will be considered index files. If not, they will be considered contents files. See Readme for a more extensive explanation.
desc_id	Logical, defaults to FALSE. If TRUE, results are returned with highest id first.
batch	An integer, defaults to NULL. If not given, a check is performed in the database to find if previous downloads have taken place. If so, by default, the current batch will be one unit higher than the highest batch number found in the database.
download_again_if_status_is_not	Defaults to NULL. If given, it must a status code as integer, typically 200L, or c(200L, 404L).
...	Arguments passed on to cas_get_urls_df, cas_get_base_folder custom_path Defaults to NULL. If given, all other parameters and settings are ignored, and folder is set to this value.

Value

A data frame with four columns: id, url, path and type

---

cas_get_options *Get key project parameters that determine the folder used for storing project files*

Description

Get key project parameters that determine the folder used for storing project files

Usage

```
cas_get_options(
  project = NULL,
  website = NULL,
  use_db = NULL,
  base_folder = NULL,
  db_type = NULL,
```

```
  db_folder = NULL,
  ...
)
```

Arguments

project	Defaults to NULL. Project name, can be set once per session with cas_set_options(). This will be used as first level folder and may be used elsewhere to describe the dataset.
website	Defaults to NULL. Website name, can be set once per session with cas_set_options(). This will be used as a second level folder and may be used elsewhere to describe the dataset.
use_db	Defaults to TRUE. If TRUE, stores information about the download process and extracted text in a local database.
base_folder	Defaults to NULL, can be set once per session with cas_set_options(). A path to a location used for storing html and other project files. If the folder does not exist, it will be created. If not given, and not previously set as environment variable, defaults to castarter_data.
db_folder	Defaults to NULL. can be set once per session with cas_set_options() or cas_set_db_folder(). A path to a location used for storing the database. If not given, and not previously set as environment variable, defaults to castarter_data.

Value

A list object with the given or previously set options.

See Also

Other settings: cas_set_options()

Examples

```
# it is possible to set only a few options, and let others be added when calling functions
cas_set_options(base_folder = fs::path(fs::path_temp(), "castarter_data"))
cas_options_list <- cas_get_options()
cas_options_list

cas_options_list2 <- cas_get_options(project = "test_project")
cas_options_list2

cas_set_options(
  base_folder = fs::path(fs::path_temp(), "castarter_data"),
  project = "test_project",
  website = "test_website"
)

cas_options_list3 <- cas_get_options()
cas_options_list3

# Passing an argument overwrites the arguments set with options
```

```
cas_options_list4 <- cas_get_options(website = "test_website4")
cas_options_list4
```

`cas_get_path_to_files` *Get path to locally downloaded files*

Description

This function relies on data stored in the database.

Usage

```
cas_get_path_to_files(
  urls = NULL,
  id = NULL,
  batch = "latest",
  status = 200,
  index = FALSE,
  index_group = NULL,
  custom_folder = NULL,
  custom_path = NULL,
  file_format = "html",
  sample = FALSE,
  db_connection = NULL,
  db_folder = NULL,
  disconnect_db = TRUE,
  ...
)
```

Arguments

`batch`	Default to "latest": returns only the path to the file with the highest batch identifier available. Valid values are: "latest", "all", or a numeric identifier corresponding to desired batch.
`status`	Defaults to 200. Keeps only files downloaded with the given status (can be more than one, given as a vector). If NULL, no filter based on status is applied.
`index`	Logical, defaults to FALSE. If TRUE, downloaded files will be considered `index` files. If not, they will be considered `contents` files. See Readme for a more extensive explanation.
`sample`	Defaults to FALSE. If TRUE, the download order is randomised. If a numeric is given, the download order is randomised and at most the given number of items is downloaded.
`db_connection`	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
`...`	Passed to `cas_get_db_file()`.

Value

A data frame of one row if "batch" is set to "latest". Possibly more than one row in other cases.

`cas_get_urls_df` *Checks that a given input corresponds to the format expected of a download data frame, consistently returns expected format*

Description

Checks that a given input corresponds to the format expected of a download data frame, consistently returns expected format

Usage

```
cas_get_urls_df(urls = NULL, index = FALSE, index_group = NULL, ...)
```

Arguments

`url`	A character vector or a data frame with at least two columns, `id` and `url`

Value

Consistently returns a data frame with at least two columns: a numeric `id` column, and a character `url` column.

Examples

```
cas_get_urls_df(c(
  "https://example.com/a/",
  "https://example.com/b/"
))
```

`cas_get_website_folder` *Get folder were files and data related to the current website are stored*

Description

Get folder were files and data related to the current website are stored

Usage

```
cas_get_website_folder(base_folder = NULL, project = NULL, website = NULL)
```

Arguments

base_folder	Defaults to NULL, can be set once per session with cas_set_options(). A path to a location used for storing html and other project files. If the folder does not exist, it will be created. If not given, and not previously set as environment variable, defaults to castarter_data.
project	Defaults to NULL. Project name, can be set once per session with cas_set_options(). This will be used as first level folder and may be used elsewhere to describe the dataset.
website	Defaults to NULL. Website name, can be set once per session with cas_set_options(). This will be used as a second level folder and may be used elsewhere to describe the dataset.

Value

A path to a folder.

Examples

```
cas_get_website_folder()
```

---

cas_ia_check *Gets an Archive.org Wayback Machine URL*

Description

For details on API access to the Wayback Machine see: https://archive.org/help/wayback_api.php

Usage

```
cas_ia_check(
  url = NULL,
  wait = 1,
  retry_times = 3,
  pause_base = 2,
  pause_cap = 512,
  pause_min = 4,
  db_connection = NULL,
  disconnect_db = FALSE,
  check_db = TRUE,
  write_db = TRUE,
  output_only_newly_checked = FALSE,
  ...
)
```

Arguments

`url`	A charachter vector of length one, a url.
`wait`	Defaults to 1. Number of seconds to wait between downloading one page and the next. Can be increased to reduce server load, or can be set to 0 when this is not an issue.
`retry_times`	Defaults to 3. Number of times to retry download in case of errors.
`check_db`	Defaults to TRUE. If TRUE, checks if given URL has already been checked in local database, and queries APIs only for URLs that have not been previously checked.
`write_db`	Defaults to TRUE. If TRUE, writes result to a local database.
`...`	Passed to `cas_get_db_file()`.

Details

For an R package facilitating more extensive interaction with the API, see: https://github.com/hrbrmstr/wayback

Integration with Wayback CDX Server API to be considered.

Value

A url linking to the version on the Internet Archive

`cas_ia_save` *Save a URL the Internet Archive's Wayback Machine*

Description

Consider using long waiting times, and using a high number of retry. Retry is done graciously, using `httr::RETRY`, and respecting the waiting time given when error 529 "too many requests" is returned by the server. This is still likely to take a long amount of time.

Usage

```
cas_ia_save(
  url = NULL,
  wait = 32,
  retry_times = 3,
  pause_base = 16,
  pause_cap = 1024,
  pause_min = 64,
  only_if_unavailable = TRUE,
  ia_check = TRUE,
  ia_check_wait = 2,
  db_connection = NULL,
  check_db = TRUE,
  write_db = TRUE,
  ...
)
```

Arguments

`url`	A charachter vector of length one, a url.
`wait`	Defaults to 32. I have found no information online about what wait time is considered suitable by Archive.org itself, but I've noticed that with wait time shorter than 10 seconds the whole process stops getting positive replies from the server very soon.
`retry_times`	Defaults to 3. Number of times to retry download in case of errors.
`pause_base, pause_cap`	This method uses exponential back-off with full jitter - this means that each request will randomly wait between `pause_min` and `pause_base * 2 ^ attempt` seconds, up to a maximum of `pause_cap` seconds.
`pause_min`	Minimum time to wait in the backoff; generally only necessary if you need pauses less than one second (which may not be kind to the server, use with caution!).
`only_if_unavailable`	Defaults to TRUE. If TRUE, checks for availability of urls before attempting to save them.
`ia_check`	Defaults to TRUE. If TRUE, checks again the URL after saving it and keeps record in the local database.
`ia_check_wait`	Defaults to 2, passed to `cas_ia_check()`. Can generally be kept low, as this is a light API.
`check_db`	Defaults to TRUE. If TRUE, checks if given URL has already been checked in local database, and queries APIs only for URLs that have not been previously checked.
`write_db`	Defaults to TRUE. If TRUE, writes result to a local database.
`...`	Passed to `cas_get_db_file()`.

Examples

```
## Not run:
if (interactive()) {
  # Once the usual parameters are set with `cas_set_options()` it is generally
  # ok to just let it get urls from the database and let it run without any
  # additional parameter.
  cas_ia_save()
}

## End(Not run)
```

`cas_kwic` *Adds a column with n words before and after the selected pattern to see keywords in context*

Description

Adds a column with n words before and after the selected pattern to see keywords in context

Usage

```
cas_kwic(
  corpus,
  pattern,
  text = text,
  words_before = 5,
  words_after = 5,
  same_sentence = TRUE,
  period_at_end_of_sentence = TRUE,
  ignore_case = TRUE,
  regex = TRUE,
  full_words_only = FALSE,
  full_word_with_partial_match = TRUE,
  pattern_column_name = pattern
)
```

Arguments

`corpus`	A textual corpus as a data frame.
`pattern`	A pattern, typically of one or more words, to be used to break text. Should be of length 1 or length equal to the number of rows.
`text`	Defaults to text. The unquoted name of the column of the corpus data frame to be used for matching.
`words_before`	Integer, defaults to 5. Number of columns to include in the `before` column.
`words_after`	Integer, defaults to 5. Number of columns to include in the `after` column.
`same_sentence`	Logical, defaults to TRUE. If TRUE, before and after include only words found in the sentence including the matched pattern.
`period_at_end_of_sentence`	Logical, defaults to TRUE. If TRUE, a period (".") is always included at the end of a sentence. Relevant only if `same_sentence` is set to TRUE.
`ignore_case`	Defaults to TRUE.
`regex`	Defaults to TRUE. Treat pattern as regex.
`full_words_only`	Defaults to FALSE. If FALSE, pattern is counted even when it is found in the middle of a word (e.g. if FALSE, "ratio" would be counted as match in the word "irrational").
`full_word_with_partial_match`	Defaults to TRUE. If TRUE, if there is a partial match of the pattern, the `pattern` column still includes the full word where the match has been found. Relevant only when `full_words_only` is set to FALSE.
`pattern_column_name`	Defaults to 'pattern'. The unquoted name of the column to be used for the word in the output.

Value

A data frame (a tibble), with the same columns as input, plus three columns: before, pattern, and after. Only rows where the pattern is found are included.

Examples

```
cas_kwic(
  corpus = tifkremlinen::kremlin_en,
  pattern = c("china", "india")
)
```

`cas_kwic_single_pattern` *Adds a column with n words before and after the selected pattern to see keywords in context*

Description

Adds a column with n words before and after the selected pattern to see keywords in context

Usage

```
cas_kwic_single_pattern(
  corpus,
  pattern,
  text = text,
  words_before = 5,
  words_after = 5,
  same_sentence = TRUE,
  period_at_end_of_sentence = TRUE,
  ignore_case = TRUE,
  regex = TRUE,
  full_words_only = FALSE,
  full_word_with_partial_match = TRUE,
  pattern_column_name = pattern
)
```

Arguments

`corpus`	A textual corpus as a data frame.
`pattern`	A pattern, typically of one or more words, to be used to break text. Should be of length 1 or length equal to the number of rows.
`text`	Defaults to text. The unquoted name of the column of the corpus data frame to be used for matching.
`words_before`	Integer, defaults to 5. Number of columns to include in the `before` column.
`words_after`	Integer, defaults to 5. Number of columns to include in the `after` column.

same_sentence Logical, defaults to TRUE. If TRUE, before and after include only words found in the sentence including the matched pattern.

period_at_end_of_sentence
Logical, defaults to TRUE. If TRUE, a period (".") is always included at the end of a sentence. Relevant only if same_sentence is set to TRUE.

ignore_case Defaults to TRUE.

regex Defaults to TRUE. Treat pattern as regex.

full_words_only
Defaults to FALSE. If FALSE, pattern is counted even when it is found in the middle of a word (e.g. if FALSE, "ratio" would be counted as match in the word "irrational").

full_word_with_partial_match
Defaults to TRUE. If TRUE, if there is a partial match of the pattern, the pattern column still includes the full word where the match has been found. Relevant only when full_words_only is set to FALSE.

pattern_column_name
Defaults to 'pattern'. The unquoted name of the column to be used for the word in the output.

Value

A data frame (a tibble), with the same columns as input, plus three columns: before, pattern, and after. Only rows where the pattern is found are included.

Examples

```
cas_kwic_single_pattern(
  corpus = tifkremlinen::kremlin_en,
  pattern = "West"
)
```

---

cas_read_corpus *Read datasets created with* cas_write_dataset

---

Description

Read datasets created with cas_write_dataset

Usage

```
cas_read_corpus(
  ...,
  update = FALSE,
  path = NULL,
  file_format = "parquet",
  partition = NULL,
  token = "full_text",
  corpus_folder = "corpus"
)
```

Arguments

...	Passed to cas_get_db_file().
update	Logical, defaults to FALSE. If FALSE, just checks if relevant corpus has been previously stored. If TRUE, it checks if more recent contents are available in the local database.
path	Defaults to NULL. If NULL, path is set to the project/website/export/dataset/file_format folder.
file_format	Defaults to "parquet". Currently, other options are not implemented.
partition	Defaults to NULL. If NULL, the parquet file is not partitioned. "year" is a common alternative: if set to "year", the parquet file is partitioned by year. If a year column does not exist, it is created based on the assumption that a date column exists and it is (or can be coerced to) a vector of class Date.
token	Defaults to "full_text", which does not tokenise the text column. If different from full_text, it is passed to tidytext::unnest_tokens (see its help for details). Accepted values include "words", "sentences", and "paragraphs". See ?tidytext::unnest_tokens() for details.

Value

A dataset as ArrowObject

Examples

```
## Not run:
cas_read_corpus()

## End(Not run)
```

---

cas_read_db_contents_data

Read contents data from local database

Description

Read contents data from local database

Usage

```
cas_read_db_contents_data(db_connection = NULL, db_folder = NULL, ...)
```

Arguments

db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
...	Passed to cas_get_db_file().

`cas_read_db_contents_id` *Read contents from local database*

Description

Read contents from local database

Usage

```
cas_read_db_contents_id(db_connection = NULL, db_folder = NULL, ...)
```

Arguments

`db_connection`	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
`...`	Passed to `cas_get_db_file()`.

Value

A data frame with three columns and data stored in the `contents_id` table of the local database. The data frame has zero rows if the database does not exist or no data was previously stored there.

Examples

```
cas_set_options(
  base_folder = fs::path(tempdir(), "R", "castarter_data"),
  db_folder = fs::path(tempdir(), "R", "castarter_data"),
  project = "example_project",
  website = "example_website"
)
cas_enable_db()


urls_df <- cas_build_urls(
  url = "https://www.example.com/news/",
  start_page = 1,
  end_page = 10
)

cas_write_db_contents(urls = urls_df)

cas_read_db_contents_id()
```

`cas_read_db_download` *Read index from local database*

Description

Read index from local database

Usage

```
cas_read_db_download(
  index = FALSE,
  id = NULL,
  batch = "latest",
  status = 200L,
  db_connection = NULL,
  db_folder = NULL,
  ...
)
```

Arguments

`batch`	Default to "latest": returns only the path to the file with the highest batch identifier available. Valid values are: "latest", "all", or a numeric identifier corresponding to desired batch.
`status`	Defaults to 200. Keeps only files downloaded with the given status (can be more than one, given as a vector). If NULL, no filter based on status is applied.
`db_connection`	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
`...`	Passed to `cas_get_db_file()`.

Value

A data frame with three columns and data stored in the `index_id` table of the local database. The data frame has zero rows if the database does not exist or no data was previously stored there.

Examples

```
cas_set_options(
  base_folder = fs::path(tempdir(), "R", "castarter_data"),
  db_folder = fs::path(tempdir(), "R", "castarter_data"),
  project = "example_project",
  website = "example_website"
)
cas_enable_db()


urls_df <- cas_build_urls(
```

```
  url = "https://www.example.com/news/",
  start_page = 1,
  end_page = 10
)

cas_write_db_index(urls = urls_df)

cas_read_db_index()
```

`cas_read_db_ia` *Read status on the Internet Archive of given URLs*

Description

Read status on the Internet Archive of given URLs

Usage

```
cas_read_db_ia(db_connection = NULL, db_folder = NULL, ...)
```

Arguments

`db_connection`	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
`...`	Passed to `cas_get_db_file()`.

`cas_read_db_ignore_id` *Read identifiers to be ignored from the local database*

Description

Read identifiers to be ignored from the local database

Usage

```
cas_read_db_ignore_id(
  db_connection = NULL,
  db_folder = NULL,
  index_group = NULL,
  disconnect_db = TRUE,
  ...
)
```

Value

A data frame with a single column, `id`

Examples

```
cas_set_options(
  base_folder = fs::path(tempdir(), "R", "cas_read_db_ignore_id"),
  db_folder = fs::path(tempdir(), "R", "cas_read_db_ignore_id"),
  project = "example_project",
  website = "example_website"
)
cas_enable_db()


cas_write_db_ignore_id(id = sample(x = 1:100, size = 10))

cas_read_db_ignore_id()
```

`cas_read_db_index` *Read index from local database*

Description

Read index from local database

Usage

```
cas_read_db_index(
  db_connection = NULL,
  db_folder = NULL,
  index_group = NULL,
  ...
)
```

Arguments

`db_connection`	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
`...`	Passed to `cas_get_db_file()`.

Value

A data frame with three columns and data stored in the `index_id` table of the local database. The data frame has zero rows if the database does not exist or no data was previously stored there.

Examples

```
cas_set_options(
  base_folder = fs::path(tempdir(), "R", "castarter_data"),
  db_folder = fs::path(tempdir(), "R", "castarter_data"),
  project = "example_project",
  website = "example_website"
```

```
)
cas_enable_db()


urls_df <- cas_build_urls(
  url = "https://www.example.com/news/",
  start_page = 1,
  end_page = 10
)

cas_write_db_index(urls = urls_df)

cas_read_db_index()
```

`cas_read_db_urls` *Read urls stored in the local database*

Description

Read urls stored in the local database

Usage

```
cas_read_db_urls(
  index = FALSE,
  db_connection = NULL,
  db_folder = NULL,
  index_group = NULL,
  ...
)
```

Arguments

`db_connection`	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
`...`	Passed to `cas_get_db_file()`.

`cas_read_from_db` *Reads data from local database*

Description

Reads data from local database

Usage

```
cas_read_from_db(
  table,
  db_folder = NULL,
  db_connection = NULL,
  disconnect_db = FALSE,
  ...
)
```

Arguments

table	Name of the table. See readme for details.
db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
disconnect_db	Defaults to TRUE. If FALSE, leaves the connection to database open.
...	Passed to cas_get_db_file().

See Also

Other database functions: cas_check_db_folder(), cas_check_use_db(), cas_connect_to_db(), cas_create_db_folder(), cas_disable_db(), cas_disconnect_from_db(), cas_enable_db(), cas_get_db_settings(), cas_set_db(), cas_set_db_folder(), cas_write_to_db()

Examples

```
cas_set_options(
  base_folder = fs::path(tempdir(), "R", "castarter_data"),
  project = "example_project",
  website = "example_website"
)
cas_enable_db()


urls_df <- cas_build_urls(
  url = "https://www.example.com/news/",
  start_page = 1,
  end_page = 10
)

cas_write_to_db(
  df = urls_df,
  table = "index_id"
)

cas_read_from_db(table = "index_id")
```

cas_reset_db *Delete a specific table from database*

Description

Delete a specific table from database

Usage

```
cas_reset_db(
  table,
  db_connection = NULL,
  disconnect_db = FALSE,
  db_folder = NULL,
  ask = TRUE,
  ...
)
```

Arguments

table	Name of the table. Yuu can use DBI::dbListTables(cas_connect_to_db()) to see currently available tables. See vignette("castarter-database") for more information about the contents and structure of each table.
db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
disconnect_db	Defaults to TRUE. If FALSE, leaves the connection to database open.
ask	Logical, defaults to TRUE. If set to FALSE, the relevant table will be deleted without asking for confirmation from the user.
...	Passed to cas_get_db_file().

cas_reset_db_contents_data *Removes from the local database the folder where extracted data are stored*

Description

Removes from the local database the folder where extracted data are stored

Usage

```
cas_reset_db_contents_data(
  db_connection = NULL,
  db_folder = NULL,
  ask = TRUE,
  ...
)
```

Arguments

db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
ask	Logical, defaults to TRUE. If set to FALSE, the relevant table will be deleted without asking for confirmation from the user.
...	Passed to cas_get_db_file().

---

cas_reset_db_contents_id

Removes from the local database the folder where links to contents associated with their id are stored

---

Description

Removes from the local database the folder where links to contents associated with their id are stored

Usage

```
cas_reset_db_contents_id(
  db_connection = NULL,
  db_folder = NULL,
  ask = TRUE,
  ...
)
```

Arguments

db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
ask	Logical, defaults to TRUE. If set to FALSE, the relevant table will be deleted without asking for confirmation from the user.
...	Passed to cas_get_db_file().

---

cas_reset_db_ignore_id

Removes from the local database all identifiers included in the ignore list

---

Description

Removes from the local database all identifiers included in the ignore list

Usage

```
cas_reset_db_ignore_id(db_connection = NULL, db_folder = NULL, ask = TRUE, ...)
```

Arguments

db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
ask	Logical, defaults to TRUE. If set to FALSE, the relevant table will be deleted without asking for confirmation from the user.
...	Passed to cas_get_db_file().

Examples

```
cas_set_options(
  base_folder = fs::path(tempdir(), "R", "cas_reset_db_ignore_id"),
  db_folder = fs::path(tempdir(), "R", "cas_reset_db_ignore_id"),
  project = "example_project",
  website = "example_website"
)
cas_enable_db()


cas_write_db_ignore_id(id = sample(x = 1:100, size = 10))

cas_read_db_ignore_id()

cas_reset_db_ignore_id(ask = FALSE)

cas_read_db_ignore_id()
```

---

cas_reset_db_index_id *Removes from the local database the table where links to index urls are stored*

Description

Removes from the local database the table where links to index urls are stored

Usage

```
cas_reset_db_index_id(db_connection = NULL, db_folder = NULL, ask = TRUE, ...)
```

Arguments

db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
ask	Logical, defaults to TRUE. If set to FALSE, the relevant table will be deleted without asking for confirmation from the user.
...	Passed to cas_get_db_file().

`cas_reset_download_contents` *Delete all files and database records for the contents pages of the current website*

Description

Delete all files and database records for the contents pages of the current website

Usage

```
cas_reset_download_contents(
  batch = NULL,
  file_format = "html",
  db_connection = NULL,
  db_folder = NULL,
  ask = TRUE,
  ...
)
```

Arguments

`batch`	Defaults to NULL. If given, only files and records related to the given batch are removed. If not given, all contents files are removed.
`db_connection`	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
`ask`	Logical, defaults to TRUE. If set to FALSE, the relevant table will be deleted without asking for confirmation from the user.
`...`	Passed to `cas_get_db_file()`.

`cas_reset_download_index` *Delete all files and database records for the index pages of the current website*

Description

Delete all files and database records for the index pages of the current website

Usage

```
cas_reset_download_index(
  batch = NULL,
  file_format = "html",
  db_connection = NULL,
  db_folder = NULL,
  ask = TRUE,
  ...
)
```

Arguments

batch	Defaults to NULL. If given, only files and records related to the given batch are removed. If not given, all index files are removed.
db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
ask	Logical, defaults to TRUE. If set to FALSE, the relevant table will be deleted without asking for confirmation from the user.
...	Passed to `cas_get_db_file()`.

---

cas_restore *Restore files from compressed files*

Description

Restore files from compressed files

Usage

```
cas_restore(
  restore_to = NULL,
  restore_from = NULL,
  file_format = "tar.gz",
  index = FALSE,
  contents = FALSE,
  batch = NULL,
  db_connection = NULL,
  db_folder = NULL,
  ...
)
```

Arguments

restore_to	Path to archive directory, defaults to NULL. If NULL, path is set to the project/website/archive folder.

restore_from	Path to archive directory, defaults to NULL. If NULL, path is set to the project/website/archive folder.
file_format	Defaults to "tar.gz", to ensure cross-platform compatibility. No other formats are supported at this stage.
db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
...	Passed to cas_get_db_file().

Value

A path to the base folder where files are stored. Corresponds to restore_to if given, or to a temporary folder if restore_to is set to NULL.

---

cas_set_db *Set database connection settings for the session*

Description

Set database connection settings for the session

Usage

```
cas_set_db(
  db_settings = NULL,
  driver = NULL,
  host = NULL,
  port,
  database,
  user,
  pwd
)
```

Arguments

db_settings	A list of database connection settings (see example)
driver	A database driver. Common database drivers include MySQL, PostgreSQL, and MariaDB. See unique(odbc::odbcListDrivers()[[1]]) for a list of locally available drivers.
host	Host address, e.g. "localhost".
port	Port to use to connect to the database.
database	Database name.
user	Database user name.
pwd	Password for the database user.

Value

A list with all given parameters (invisibly).

See Also

Other database functions: cas_check_db_folder(), cas_check_use_db(), cas_connect_to_db(), cas_create_db_folder(), cas_disable_db(), cas_disconnect_from_db(), cas_enable_db(), cas_get_db_settings(), cas_read_from_db(), cas_set_db_folder(), cas_write_to_db()

Examples

```
if (interactive()) {
  # Settings can be provided either as a list
  db_settings <- list(
    driver = "MySQL",
    host = "localhost",
    port = 3306,
    database = "castarter",
    user = "secret_username",
    pwd = "secret_password"
  )

  cas_set_db(db_settings)

  # or as parameters

  cas_set_db(
    driver = "MySQL",
    host = "localhost",
    port = 3306,
    database = "castarter",
    user = "secret_username",
    pwd = "secret_password"
  )
}
```

---

cas_set_db_folder *Set folder for storing the database*

Description

Consider using a folder out of your current project directory, e.g. `cas_set_db_folder("~/R/cas_data/")`: you will be able to use the same database in different projects, and prevent database files from being sync-ed if you use services such as Nextcloud or Dropbox.

Usage

```
cas_set_db_folder(path = NULL, ...)

cas_get_db_folder(path = NULL, ...)
```

Arguments

path	A path to a location used for storing the database. If the folder does not exist, it will be created.

Value

The path to the database folder, if previously set; the same path as given to the function; or the default, cas_data is none is given.

See Also

Other database functions: cas_check_db_folder(), cas_check_use_db(), cas_connect_to_db(), cas_create_db_folder(), cas_disable_db(), cas_disconnect_from_db(), cas_enable_db(), cas_get_db_settings(), cas_read_from_db(), cas_set_db(), cas_write_to_db()

Examples

```
cas_set_db_folder(fs::path(fs::path_home_r(), "R", "cas_data"))

cas_set_db_folder(fs::path(fs::path_temp(), "cas_data"))
cas_get_db_folder()
```

---

cas_set_options *Set key project parameters that determine the folder used for storing project files*

Description

Your project folder can be anywhwere on your file system. Considering that this is where possibly a very large number of html files will be downloaded, it is usually preferrable to choose a location that is not included in live backups. These settings determine the names given to these hierarchical folders: website folder will be under project folder which will be under the base_folder.

Usage

```
cas_set_options(
  project = NULL,
  website = NULL,
  use_db = TRUE,
  base_folder = NULL,
  db_type = "SQLite",
  db_folder = NULL
)
```

Arguments

project	Defaults to NULL. Project name, can be set once per session with cas_set_options(). This will be used as first level folder and may be used elsewhere to describe the dataset.
website	Defaults to NULL. Website name, can be set once per session with cas_set_options(). This will be used as a second level folder and may be used elsewhere to describe the dataset.
use_db	Defaults to TRUE. If TRUE, stores information about the download process and extracted text in a local database.
base_folder	Defaults to NULL, can be set once per session with cas_set_options(). A path to a location used for storing html and other project files. If the folder does not exist, it will be created. If not given, and not previously set as environment variable, defaults to castarter_data.
db_folder	Defaults to NULL. can be set once per session with cas_set_options() or cas_set_db_folder(). A path to a location used for storing the database. If not given, and not previously set as environment variable, defaults to castarter_data.

Value

Nothing, used for its side effects (setting options).

See Also

Other settings: cas_get_options()

Examples

```
cas_set_options(base_folder = fs::path(fs::path_temp(), "castarter_data"))
cas_options_list <- cas_get_options()
cas_options_list
```

---

cas_show_barchart_ggiraph

Creates interacative barchart with ggiraph

Description

For detail on parameters, see https://davidgohel.github.io/ggiraph/articles/offcran/using_ggiraph.html

Usage

```
cas_show_barchart_ggiraph(
  ggobj,
  data_id = NULL,
  tooltip = NULL,
  position = "stack"
)
```

Arguments

ggobj	A ggplot2 object, typically generated with cas_show_gg_base()
data_id	Defaults to NULL. If given, unquoted name of column, passed to ggiraph.
tooltip	Defaults to NULL. If given, unquoted name of column, passed to ggiraph.
position	Defaults to "stack". Available values include "dodge".

Value

A girafe/htmlwidget object

---

cas_show_barchart_ggplot2

Creates barchart with ggplot2

Description

Creates barchart with ggplot2

Usage

```
cas_show_barchart_ggplot2(ggobj, position = "stack")
```

Arguments

ggobj	A ggplot2 object, typically generated with cas_show_gg_base()
position	Defaults to "dodge". Available values include "stack".

Value

A ggplot2 object.

Examples

```
cas_count(corpus = tifkremlinen::kremlin_en,
pattern = c("putin", "medvedev")) |>
  cas_summarise(period = "year") |>
  cas_show_gg_base() |>
  cas_show_barchart_ggplot2(position = "stack")
```

cas_show_gg_base *Creates base ggplot2 object to be used by ggplot or ggiraph*

Description

Creates base ggplot2 object to be used by ggplot or ggiraph

Usage

```
cas_show_gg_base(
  count_df,
  group_by = date,
  n_column_name = n,
  pattern_column_name = pattern,
  group_as_factor = FALSE,
  font_base_size = 14
)
```

Arguments

group_by	Defaults to NULL. If given, the unquoted name of the column to be used for grouping (e.g. date, or doc_id, or source, etc.)
n_column_name	Defaults to 'n'. The unquoted name of the column to be used for the count in the output.
pattern_column_name	Defaults to 'pattern'. The unquoted name of the column to be used for the word in the output.
group_as_factor	Defaults to FALSE. If TRUE, the grouping column is forced into a factor, otherwise it is kept in its current format (e.g. date, or numeric).

Value

A ggplot2 object with aesthetics set, but no geometry.

Examples

```
cas_count(corpus = tifkremlinen::kremlin_en,
pattern = c("putin", "medvedev")) |>
  cas_summarise(period = "year") |>
  cas_show_gg_base() |>
  cas_show_barchart_ggplot2(position = "dodge")
```

cas_show_ts_dygraph *Create dygraphs based on a data frame typically generated with cas_count()*

Description

Create dygraphs based on a data frame typically generated with cas_count()

Usage

```
cas_show_ts_dygraph(
  count_df,
  date_column_name = date,
  n_column_name = n,
  pattern_column_name = pattern,
  range_selector = TRUE
)
```

Arguments

```
count_df
```

Examples

```
count_df <- castarter::cas_count(
  corpus = castarter::cas_demo_corpus,
  words = c("russia", "moscow")
) %>%
  cas_summarise(before = 15, after = 15)
cas_show_ts_dygraph(count_df)
```

cas_summarise *Summarise for a given time period word counts, typically calculatd with* `cas_count()`

Description

Summarise for a given time period word counts, typically calculatd with `cas_count()`

Usage

```
cas_summarise(
  count_df,
  date_column_name = date,
  n_column_name = n,
  pattern_column_name = pattern,
```

```
  period = NULL,
  f = mean,
  period_summary_function = sum,
  every = 1L,
  before = 0L,
  after = 0L,
  complete = FALSE,
  auto_convert = FALSE
)
```

Arguments

count_df	A data frame. Must include at least a column with a date or date-time column and a column with number of occurrences for the given time.
period	Defaults to NULL. A string describing the time unit to be used for summarising. Possible values include "year", "quarter", "month", "day", "hour", "minute", "second", "millisecond".
f	Defaults to mean. Function to be applied over n for all the values in a given time period. Common alternatives would be mean or median.
period_summary_function	Defaults to sum. This is applied when grouping by period (e.g. when period is set to year). When calculating absolute word frequency, the default (sum) is fine. When calculating relative frequencies, then mean would be more appropriate, but extra consideration should be given to the implications if then a rolling average is applied.
every	[positive integer(1)] The number of periods to group together. For example, if the period was set to "year" with an every value of 2, then the years 1970 and 1971 would be placed in the same group.
before, after	[integer(1) / Inf] The number of values before or after the current element to include in the sliding window. Set to Inf to select all elements before or after the current element. Negative values are allowed, which allows you to "look forward" from the current element if used as the .before value, or "look backwards" if used as .after.
complete	[logical(1)] Should the function be evaluated on complete windows only? If FALSE, the default, then partial computations will be allowed.
auto_convert	Defaults to FALSE. If FALSE, the date column is returned using the same format as the input; the minimun vale in the given group is used for reference (e.g. all values for January 2022 are summarised as 2021-01-01 it the data were originally given as dates.). If TRUE, it tries to adapt the output to the most intuitive correspondent type; for year, a numeric column with only the year number, for quarter in the format 2022.1, for month in the format 2022-01.
date	Defaults to date. Unquoted name of a column having either date or date-time as class.

n Unquoted to n. Unquoted name of a column having number of occurrences per time unit.

Value

A data frame with two columns: the name of the period, and the same name originally used for n.

Examples

```
## Not run:
# this assumes dates are provided in a column called date
corpus_df %>%
  cas_count(
    pattern = "example",
    group_by = date
  ) %>%
  cas_summarise(period = "year")

## End(Not run)
```

---

cas_update *Update corpus*

Description

Currently supports only update when re-downloading index urls is expected to bring new articles. It takes the first urls for each index group, and continues downloading new index pages as long as new links are found in each page. If no new link is found, it stops downloading and moves to the next index group.

Usage

```
cas_update(
  extract_links_partial,
  extractors,
  post_processing = NULL,
  wait = 3,
  user_agent = NULL,
  ...
)
```

Arguments

extract_links_partial
: A partial function, typically created with purrr::partial(.f = cas_extract_links), followed by the paramters originally used by cas_extract_links(). See examples.

extractors	A named list of functions. See examples for details.
post_processing	Defaults to NULL. If given, it must be a function that takes a data frame as input (logically, a row of the dataset) and returns it with additional or modified columns.
wait	Defaults to 1. Number of seconds to wait between downloading one page and the next. Can be increased to reduce server load, or can be set to 0 when this is not an issue.
user_agent	Defaults to NULL. If given, passed to download method.
...	Passed to `cas_get_db_file()`.

Examples

```
# Example of extract_links_partial:
extract_links_partial <- purrr::partial(
  .f = cas_extract_links,
  reverse_order = TRUE,
  container = "div",
  container_class = "hentry h-entry hentry_event",
  exclude_when = c("/photos", "/videos"),
  domain = "http://en.kremlin.ru/"
)
```

---

cas_write_corpus *Export the textual dataset for the current website*

Description

Export the textual dataset for the current website

Usage

```
cas_write_corpus(
  corpus = NULL,
  to_lower = FALSE,
  drop_na = TRUE,
  drop_empty = TRUE,
  date = date,
  text = text,
  tif_compliant = FALSE,
  file_format = "parquet",
  partition = NULL,
  token = "full_text",
  corpus_folder = "corpus",
  path = NULL,
  db_connection = NULL,
```

```
  db_folder = NULL,
  ...
)
```

Arguments

corpus	Defaults to NULL. If NULL, retrieves corpus from the current website with `cas_read_db_contents_data()`. If given, it is expected to be a corresponding data frame.
to_lower	Defaults to FALSE. Whether to convert tokens to lowercase. Passed to `tidytext` if token is not `full_text`.
drop_na	Defaults to TRUE. If TRUE, items that have NA in their `text` or `date` columns are dropped. This is often useful, as in many cases these may have other issues and/or cause inconsistencies in further analyses.
drop_empty	Defaults to TRUE. If TRUE, items that have empty elements ("") in their `text` or `date` columns are dropped. This is often useful, as in many cases these may have other issues and/or cause inconsistencies in further analyses.
date	Unquoted date column, defaults to `date`.
text	Unquoted text column, defaults to `text`. If `tif_compliant` is set to TRUE, it will be renamed to "text" even if originally it had a different name.
tif_compliant	Defaults to FALSE. If TRUE, it ensures that the first column is a character vector named "doc_id" and that the second column is a character vector named "text". See https://docs.ropensci.org/tif/ for details
file_format	Defaults to "parquet". Currently, other options are not implemented.
partition	Defaults to NULL. If NULL, the parquet file is not partitioned. "year" is a common alternative: if set to "year", the parquet file is partitioned by year. If a `year` column does not exist, it is created based on the assumption that a `date` column exists and it is (or can be coerced to) a vector of class `Date`.
token	Defaults to "full_text", which does not tokenise the text column. If different from `full_text`, it is passed to `tidytext::unnest_tokens` (see its help for details). Accepted values include "words", "sentences", and "paragraphs". See `?tidytext::unnest_tokens()` for details.
path	Defaults to NULL. If NULL, path is set to the project/website/export/dataset/file_format folder.
db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
...	Passed to `cas_get_db_file()`.

`cas_write_db_contents_data` *Write extracted contents to local database*

Description

If some IDs are already present in the database, only the new ones are appended: IDs are expected to be unique.

Usage

```
cas_write_db_contents_data(
  contents_df,
  overwrite = FALSE,
  db_connection = NULL,
  disconnect_db = FALSE,
  quiet = FALSE,
  check_previous = TRUE,
  ...
)
```

Arguments

`overwrite`	Logical, defaults to FALSE. If TRUE, checks if matching data are previously held in the table and overwrites them. This should be used with caution, as it may overwrite completely the selected table.
`db_connection`	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
`disconnect_db`	Defaults to TRUE. If FALSE, leaves the connection to database open.
`quiet`	Defaults to FALSE. If set to TRUE, messages on number of lines added are not shown.
`check_previous`	Defaults to TRUE. If set to FALSE, the given input is stored in the database without checking if the same id had already been stored.
`...`	Passed to `cas_get_db_file()`.

Details

Check for consistency in database columns: if new columns do not match previous columns, it throws an error.

Value

Invisibly returns only new rows added.

`cas_write_db_contents_id` *Write contents URLs to local database*

Description

If some URLs are already included in the database, it appends only the new ones: URLs are expected to be unique.

Usage

```
cas_write_db_contents_id(
  urls,
  overwrite = FALSE,
  db_connection = NULL,
  disconnect_db = FALSE,
  quiet = FALSE,
  check_previous = TRUE,
  ...
)
```

Arguments

`urls`	A data frame with five columns, such as `casdb_empty_contents_id`, or a character vector.
`overwrite`	Logical, defaults to FALSE. If TRUE, checks if matching data are previously held in the table and overwrites them. This should be used with caution, as it may overwrite completely the selected table.
`db_connection`	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
`disconnect_db`	Defaults to TRUE. If FALSE, leaves the connection to database open.
`quiet`	Defaults to FALSE. If set to TRUE, messages on number of lines added are not shown.
`check_previous`	Defaults to TRUE. If set to FALSE, the given input is stored in the database without checking if the same url had already been stored.
`...`	Passed to `cas_get_db_file()`.

Value

Invisibly returns only new rows added.

Examples

```
cas_set_options(
  base_folder = fs::path(tempdir(), "R", "castarter_data"),
  db_folder = fs::path(tempdir(), "R", "castarter_data"),
  project = "example_project",
  website = "example_website"
)
cas_enable_db()


urls_df <- cas_build_urls(
  url = "https://www.example.com/news/",
  start_page = 1,
  end_page = 10
)

cas_write_db_contents_id(urls = urls_df)

cas_read_db_contents_id()
```

---

cas_write_db_ignore_id

Ignore a set of ids from the download or processing step

Description

There are two main use cases for this function:

- a number of the files downloaded turned out to be irrelevant. Rather than delete any trace about them, it may be preferrable to just ignore them, so they are not processed when extracting data.
- urls originally included for download, but not yet downloaded, should be ignored and not downloaded. This may or may not be a temporary arrangement, but it is considered useful to keep the urls in the database.

Usage

```
cas_write_db_ignore_id(
  id,
  db_folder = NULL,
  db_connection = NULL,
  disconnect_db = FALSE,
  ...
)

cas_ignore_id(
  id,
  db_folder = NULL,
  db_connection = NULL,
```

```
  disconnect_db = FALSE,
  ...
)
```

Arguments

id	Defaults to NULL. A vector of id. Rows with the given id will be added to the ignore table.

Examples

```
cas_set_options(
  base_folder = fs::path(tempdir(), "R", "cas_write_db_ignore_id"),
  db_folder = fs::path(tempdir(), "R", "cas_write_db_ignore_id"),
  project = "example_project",
  website = "example_website"
)
cas_enable_db()


cas_write_db_ignore_id(id = sample(x = 1:100, size = 10))

cas_read_db_ignore_id()
```

---

cas_write_db_index *Write index URLs to local database*

Description

If some URLs are already included in the database, it appends only the new ones: URLs are expected to be unique.

Usage

```
cas_write_db_index(
  urls,
  overwrite = FALSE,
  db_connection = NULL,
  disconnect_db = FALSE,
  ...
)
```

Arguments

urls	A data frame with three columns, with the same name and type as `casdb_empty_index_id`, or a character vector.
overwrite	Logical, defaults to FALSE. If TRUE, checks if matching data are previously held in the table and overwrites them. This should be used with caution, as it may overwrite completely the selected table.

db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
disconnect_db	Defaults to TRUE. If FALSE, leaves the connection to database open.
...	Passed to cas_get_db_file().

Value

Invisibly returns only new rows added.

Examples

```
cas_set_options(
  base_folder = fs::path(tempdir(), "R", "castarter_data"),
  db_folder = fs::path(tempdir(), "R", "castarter_data"),
  project = "example_project",
  website = "example_website"
)
cas_enable_db()


urls_df <- cas_build_urls(
  url = "https://www.example.com/news/",
  start_page = 1,
  end_page = 10
)

cas_write_db_index(urls = urls_df)

cas_read_db_index()
```

---

cas_write_db_urls *Write index or contents urls directly to the local database*

Description

Write index or contents urls directly to the local database

Usage

```
cas_write_db_urls(
  urls,
  index = FALSE,
  overwrite = FALSE,
  db_connection = NULL,
  disconnect_db = FALSE,
  quiet = FALSE,
  check_previous = TRUE,
  ...
)
```

Arguments

urls	A data frame with five columns, such as casdb_empty_contents_id, or a character vector.
index	Logical, defaults to FALSE. If TRUE, downloaded files will be considered index files. If not, they will be considered contents files. See Readme for a more extensive explanation.
overwrite	Logical, defaults to FALSE. If TRUE, checks if matching data are previously held in the table and overwrites them. This should be used with caution, as it may overwrite completely the selected table.
db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
disconnect_db	Defaults to TRUE. If FALSE, leaves the connection to database open.
quiet	Defaults to FALSE. If set to TRUE, messages on number of lines added are not shown.
check_previous	Defaults to TRUE. If set to FALSE, the given input is stored in the database without checking if the same url had already been stored.
...	Passed to cas_get_db_file().

---

cas_write_to_db *Generic function for writing to database*

Description

Generic function for writing to database

Usage

```
cas_write_to_db(
  df,
  table,
  overwrite = FALSE,
  db_connection = NULL,
  disconnect_db = FALSE,
  ...
)
```

Arguments

df	A data frame. Must correspond with the type of data expected for each table.
table	Name of the table. See readme for details.
overwrite	Logical, defaults to FALSE. If TRUE, checks if matching data are previously held in the table and overwrites them. This should be used with caution, as it may overwrite completely the selected table.

db_connection	Defaults to NULL. If NULL, uses local SQLite database. If given, must be a connection object or a list with relevant connection settings (see example).
disconnect_db	Defaults to TRUE. If FALSE, leaves the connection to database open.
...	Passed to cas_get_db_file().

Value

If successful, returns invisibly the same data frame provided as input and written to the database. Returns silently NULL, if nothing is added, e.g. because use_db is set to FALSE.

See Also

Other database functions: cas_check_db_folder(), cas_check_use_db(), cas_connect_to_db(), cas_create_db_folder(), cas_disable_db(), cas_disconnect_from_db(), cas_enable_db(), cas_get_db_settings(), cas_read_from_db(), cas_set_db(), cas_set_db_folder()

Examples

```
cas_set_options(
  base_folder = fs::path(tempdir(), "R", "castarter_data"),
  project = "example_project",
  website = "example_website"
)
cas_enable_db()


urls_df <- cas_build_urls(
  url = "https://www.example.com/news/",
  start_page = 1,
  end_page = 10
)

cas_write_to_db(
  df = urls_df,
  table = "index_id"
)
```

Index